

Bebras Australia Computational Thinking Challenge

2021 Solutions Guide Round 1



Secondary School
Grades 7-12

bebras.edu.au

Bebras Australia Computational Thinking Challenge

Bebras is an international initiative aiming to promote Computational Thinking skills among students.

Started in 2004 by Professor Valentina Dagiene from the University of Vilnius, 'Bebras' is Lithuanian for beaver. This refers to their collaborative nature and strong work ethic.

The International Bebras Committee meets annually to assess potential questions and share resources. Questions are submitted by member countries and undergo a vetting process.

The Bebras international community has now grown to 60 countries with over 2.9 million students participating worldwide!

Bebras Australia began in 2014 and is now administered through CSIRO Digital Careers.

In Australia, the Bebras Challenge takes place in March and August-September each year. As of 2020, two separate challenges are offered for each round.

To find out more and register for the next challenge, visit bebras.edu.au

Engaging young minds for Australia's digital future

CSIRO Digital Careers supports teachers and encourages students' understanding of digital technologies and the foundational skills they require in an ever-changing workforce. Growing demand for digital skills isn't just limited to the ICT sector. All jobs of the future will require them, from marketing and multimedia through to agriculture, finance and health. Digital Careers prepares students with the knowledge and skills they need to thrive in the workforce of tomorrow.

digitalcareers.csiro.au

523

Australian schools participated in Bebras R1 2021



32 311

Australian students participated in Bebras R1 2021



2.9 million

students participate worldwide



digital
careers



What is a Solutions Guide?

Computational Thinking skills underpin the careers of the future. Creating opportunities for students to engage in activities that utilise their critical and creative thinking along with problem solving skills is essential to further learning. The Bebras Challenge is an engaging way for students to learn and practice these skills.

Within this Solutions Guide you will find all of the questions and tasks from Round 1 of the Bebras Australia Computational Thinking Challenge 2021. On each page above the question you will find the age group, level of difficulty, country of origin and key Computational Thinking skills.

After each question you will find the answer, an explanation, the Computational Thinking skills most commonly used, and the Australian Digital Technologies curriculum key concepts featured.

Contents

What is a Solutions Guide?	3	Years 11+12	
What is Computational Thinking?	5	Don't Crash	64
Computational Thinking alignment	6	Stickers	66
Digital Technologies key concepts	8	Lemmings	68
Digital Technologies alignment	9	Electric Cars	70
		Needlework	72
		Digital Trees	74
		Letter Code	76
		Rabbit Paddock	77
		Reversibility	79
		Musical Instrument	81
		Royal Fountains	82
		Vulnerable	83
		Marbles and Boxes	84
		Blinking LEDs	85
		Game of Whispers	87
Years 7+8			
Jumping Kangaroo	12		
Jacques the Porter	13		
Processing Objects	15		
Party Message	16		
Jigsaw Puzzle	18		
Robot Maze Game	20		
Grocery Stores	22		
Beaver vs. Kangaroo	24		
Hansel and Gretel	26		
News Travels Fast	27		
Beaver Time	28		
New Neighbour	30		
Air Conditioning	32		
Heat Maps	33		
Household Appliances	34		
Years 9+10			
Winners or Losers	37		
Magic Drink Machine	38		
Damaged Secret Table	40		
Creating Numbers	42		
Echo Cipher	43		
Sierpinski Triangle	45		
Passwords	47		
Tree Sudoku	49		
Epidemic Crisis	51		
Aggelos the Mailman	53		
Heavy Parts	54		
Math Machine	56		
Wood Processing	58		
Towers of Blocks	60		
Electric Car Queues	62		

What is Computational Thinking?

Computational Thinking is a set of skills that underpin learning within the Digital Technologies classroom. These skills allow students to engage with processes, techniques and digital systems to create improved solutions to address specific problems, opportunities or needs. Computational Thinking uses a number of skills, including:



DECOMPOSITION

Breaking down problems into smaller, easier parts.



PATTERN RECOGNITION

Using patterns in information to solve problems.



ABSTRACTION

Finding information that is useful and taking away any information that is unhelpful.



MODELLING AND SIMULATION

Trying out different solutions or tracing the path of information to solve problems.



ALGORITHMS

Creating a set of instructions for solving a problem or completing a task.



EVALUATION

Assessing a solution to a problem and using that information again on new problems.

More Computational Thinking resources

Visit digitalcareers.csiro.au/CTIA to download the Computational Thinking in Action worksheets. These can be used as discussion prompts, extension activities or a framework to build a class project.

Each resource was designed to develop teamwork; critical and creative thinking; problem solving; and Computational Thinking skills.



Computational Thinking skills alignment

2021 Round 1 Questions	Grade level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Years 7+8							
Jumping Kangaroo	Easy						
Jacques the Porter	Easy						
Processing Objects	Easy						
Party Message	Easy						
Jigsaw Puzzle	Easy						
Robot Maze Game	Medium						
Grocery Stores	Medium						
Beaver vs. Kangaroo	Medium						
Hansel and Gretel	Medium						
News Travels Fast	Medium						
Beaver Time	Hard						
New Neighbour	Hard						
Air Conditioning	Hard						
Heat Maps	Hard						
Household Appliances	Hard						
Years 9+10							
Winners or Losers	Easy						
Magic Drink Machine	Easy						
Damaged Secret Table	Easy						
Creating Numbers	Easy						
Echo Cipher	Easy						
Sierpinski Triangle	Medium						
Passwords	Medium						
Tree Sudoku	Medium						
Epidemic Crisis	Medium						
Aggelos the Mailman	Medium						
Heavy Parts	Hard						
Math Machine	Hard						
Wood Processing	Hard						
Towers of Blocks	Hard						
Electric Car Queues	Hard						

Computational Thinking skills alignment

2021 Round 1 Questions	Grade level	Decomposition	Pattern Recognition	Abstraction	Modelling & Simulation	Algorithms	Evaluation
Years 11+12							
Don't Crash	Easy						
Stickers	Easy						
Lemmings	Easy						
Electric Cars	Easy						
Needlework	Easy						
Digital Trees	Medium						
Letter Code	Medium						
Rabbit Paddock	Medium						
Reversibility	Medium						
Musical Instrument	Medium						
Royal Fountains	Hard						
Vulnerable	Hard						
Marbles and Boxes	Hard						
Blinking LEDs	Hard						
Game of Whispers	Hard						

Australian Digital Technologies curriculum key concepts

Abstraction

Hiding details of an idea, problem or solution that are not relevant, to focus on a manageable number of aspects.

Data Collection

Numerical, categorical, or structured values collected or calculated to create information, e.g. the Census.

Data Representation

How data is represented and structured symbolically for storage and communication, by people and in digital systems.

Data Interpretation

The process of extracting meaning from data. Methods include modelling, statistical analysis, and visualisation.

Specification

Defining a problem precisely and clearly, identifying the requirements, and breaking it down into manageable pieces.

Algorithms

The precise sequence of steps and decisions needed to solve a problem. They often involve iterative (repeated) processes.

Implementation

The automation of an algorithm, typically by writing a computer program (coding) or using appropriate software.

Digital Systems

A system that processes data in binary, made up of hardware, controlled by software, and connected to form networks.

Interactions

Human–Human Interactions: How users use digital systems to communicate and collaborate.

Human–Computer Interactions: How users experience and interface with digital systems.

Impact

Analysing and predicting how existing and created systems meet needs, affect people, and change society and the world.

For more information on the Digital Technologies curriculum, please visit the Australian Curriculum, Assessment and Reporting Authority (ACARA) website:
australiancurriculum.edu.au/f-10-curriculum/technologies/digital-technologies

Digital Technologies

key concepts alignment

2021 Round 1 Questions	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Years 7+8										
Jumping Kangaroo										
Jacques the Porter										
Processing Objects										
Party Message										
Jigsaw Puzzle										
Robot Maze Game										
Grocery Stores										
Beaver vs. Kangaroo										
Hansel and Gretel										
News Travels Fast										
Beaver Time										
New Neighbour										
Air Conditioning										
Heat Maps										
Household Appliances										
Years 9+10										
Winners or Losers										
Magic Drink Machine										
Damaged Secret Table										
Creating Numbers										
Echo Cipher										
Sierpinski Triangle										
Passwords										
Tree Sudoku										
Epidemic Crisis										
Aggelos the Mailman										
Heavy Parts										
Math Machine										
Wood Processing										
Towers of Blocks										
Electric Car Queues										

Digital Technologies

key concepts alignment

2021 Round 1 Questions	Abstraction	Data Collection	Data Representation	Data Interpretation	Specification	Algorithms	Implementation	Digital Systems	Interactions	Impacts
Years 11+12										
Don't Crash										
Stickers										
Lemmings										
Electric Cars										
Needlework										
Digital Trees										
Letter Code										
Rabbit Paddock										
Reversibility										
Musical Instrument										
Royal Fountains										
Vulnerable										
Marbles and Boxes										
Blinking LEDs										
Game of Whispers										

Bebras Challenge 2021 Round 1

Years 7+8

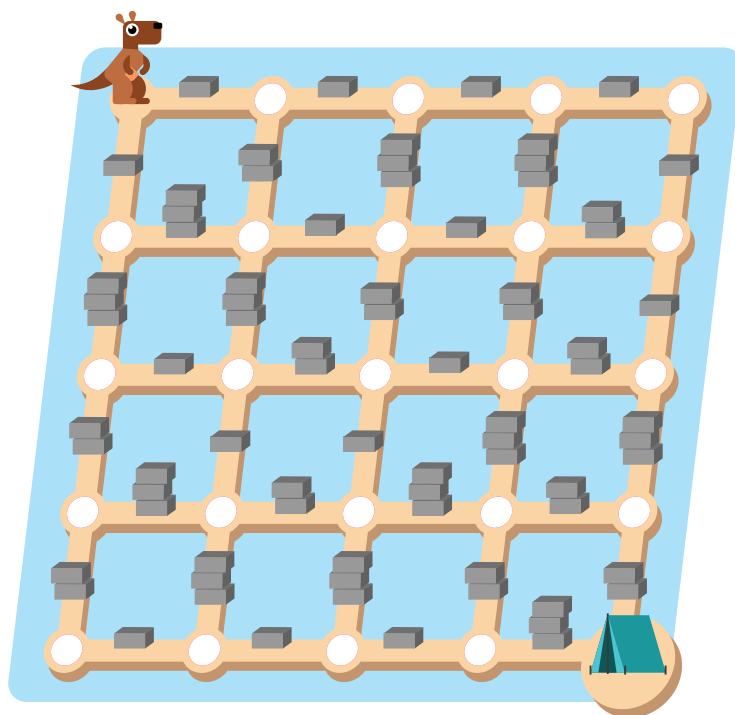
Jumping Kangaroo

A kangaroo is trying to get home. She can only jump vertically (up and down) or horizontally (left to right) along the path, and only if there are no more than two bricks stacked on the path.

The kangaroo wants to get home as quickly as possible.

Question

Show the path the kangaroo takes to get home.



Answer

The solution is shown in the picture.

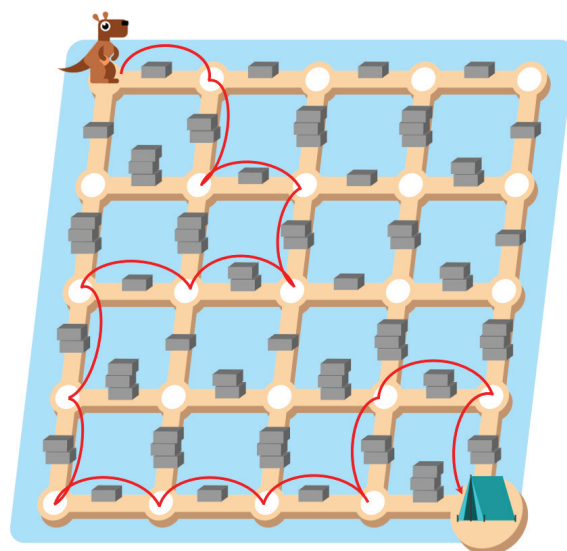
It's Computational Thinking

Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Specification, Algorithms

To find a solution to this problem, you can start the search, going step by step and, if you end up in a dead end where all possible paths have three stones, go back (possibly, by several steps) and try another sequence of possible steps.

This approach is known in Computer Science as backtracking, and it is a technique used in many algorithms. It can be used to solve puzzles or sudoku or combinatorial optimization problems.



This task shows that, sometimes, it is more efficient to start working out the solution from the “end”, here working our way back from the kangaroo’s home. We can see that here, doing so requires us to perform less backtracking, and the solution is easier to find. But without prior exploration of the problem, it is impossible to say whether it would be better to start from the beginning or from the end.



Jacques the Porter

Jacques is a porter in an apartment building. The building has five apartments, each occupied by one beaver. As the beavers leave for work, they give their keys to Jacques. To avoid mixing up the keys, Jacques uses a key locker for each apartment. On each locker are the first three letters of the owner's name.

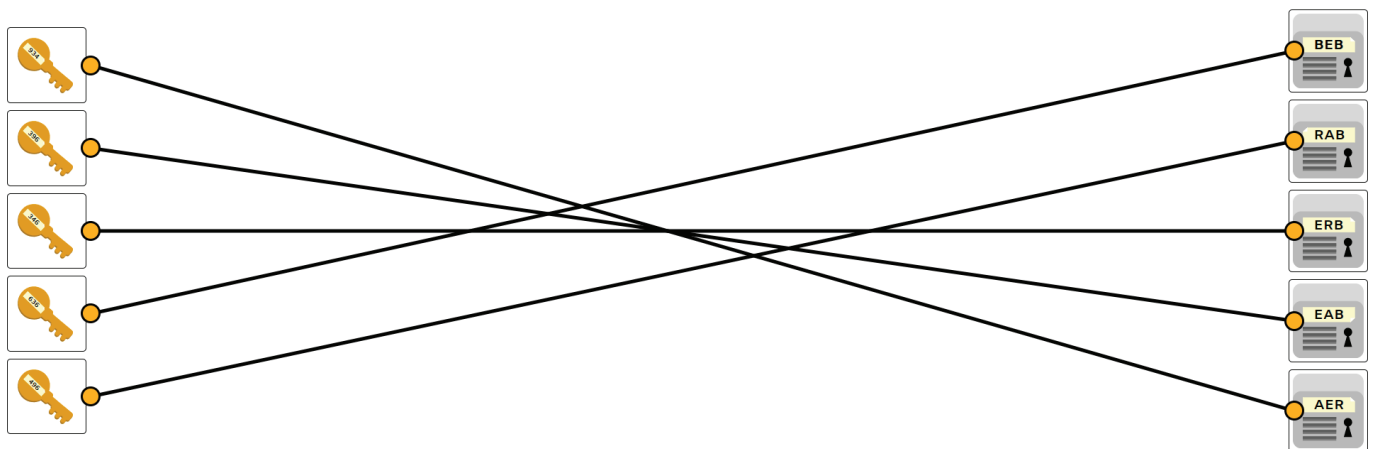
For security reasons, no names are written on the keys. Instead Jacques labelled the keys with three digits to indicate the owner. The same letter always corresponds to the same single digit across all the keys.

Question

Connect the keys to their lockers.



Answer



The locker BEB is the only one where the first and the third letter is the same, B. So this locker matches with the key 636. Now we know that the letter B corresponds to 6 and the letter E to 3. The locker AER is the only one that ends with a letter that is not B (R). Thus, the letter R corresponds to 4 and the only suitable key is 934. Based on that, the letter A corresponds to 9. Now we can match EAB with 396 and ERB with 346. At this point, the locker RAB is the only one left and the label is 496.

Continued on next page



Jacques the Porter - continued

It's Computational Thinking

Computational Thinking Skills: Decomposition, Pattern Recognition, Abstraction

Concepts: Abstraction, Data Representation, Data Interpretation

Encryption and decryption are fundamental concepts in the Computer Science discipline called cryptography. In this task the encryption process is based on the monoalphabetic substitution cipher example called the Vatsyayana cipher. The idea came from an Indian text from the 4th century AD. In this task the Vatsyayana cipher is used to assign a certain letter to a certain digit. The certain letter and a certain digit can be used only in one pair, no re-usage is allowed. To encrypt a message each letter is substituted by a paired digit. Once the message is received the decryption process starts by substituting a digit with a paired letter. Encryption and decryption use the fixed substitution over the entire message. Today, it is an unsecure method of encrypting messages as once we know a correct pair of letters and a digit we can apply this knowledge to decrypting the next piece of the message.



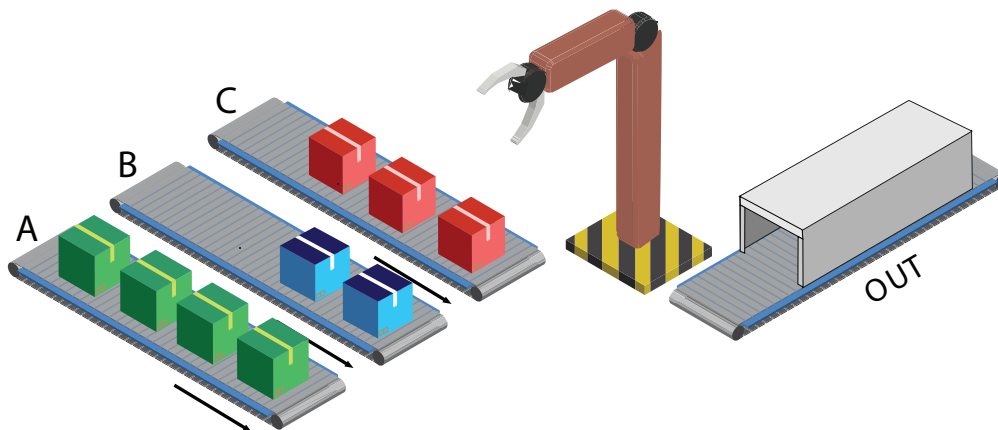
Processing Objects

A robot arm takes objects from three conveyor belts (A, B and C) and moves them to the processing conveyor belt (OUT).

Here is how the robot arm works:

- First it takes an object from A and moves it to OUT
- Then it moves to B, takes an object from B and moves it to OUT
- Finally, it does the same for C, before starting again at A

When there is no object to pick up on a conveyor belt, the robot arm waits until one becomes available, because the processing unit needs one object from each conveyor belt to proceed.



Question

Given the situation shown in the picture, and knowing that no new objects will arrive on the conveyor belts (A, B and C), how many objects will be moved by the arm?

Answer

The correct answer is 7.

Indeed, the arm will first move an object from conveyor belt A, then one from B and finally one from C (which is already 3 objects). It then goes back to A, followed by B and C (which makes now a total of 6 objects). Finally, the arm starts again with A (which makes now a total of 7 objects) and then moves to B where there is no more object. Since no new objects are coming, the arm will be stuck on B after the two complete iterations.

It's Computational Thinking

Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms

Concepts: Abstraction, Specification, Algorithms, Digital Systems

The process the robot is executing here has close ties with scheduling, that is, a method to decide how resources are assigned to jobs to be executed to perform and complete a work. Here, the robot arm has to move three objects to a processing conveyor belt that perform some work with the three objects. In order to be sure that only three objects, and only one of each, are moved to the processing conveyor belt at the same time, a specific scheduling method described in the task has been programmed in the robot arm.

The task is also about understanding an algorithm and executing it, to predict a future state. Given an initial situation (the position of the robot arm and the objects on the A, B and C conveyor belts) and an algorithm, executing it, step-by-step, to find out the produced result is an important activity that programmers have to do. It is known as debugging when they have to find a possible bug, that is, something that will go wrong in the program.

Party Message

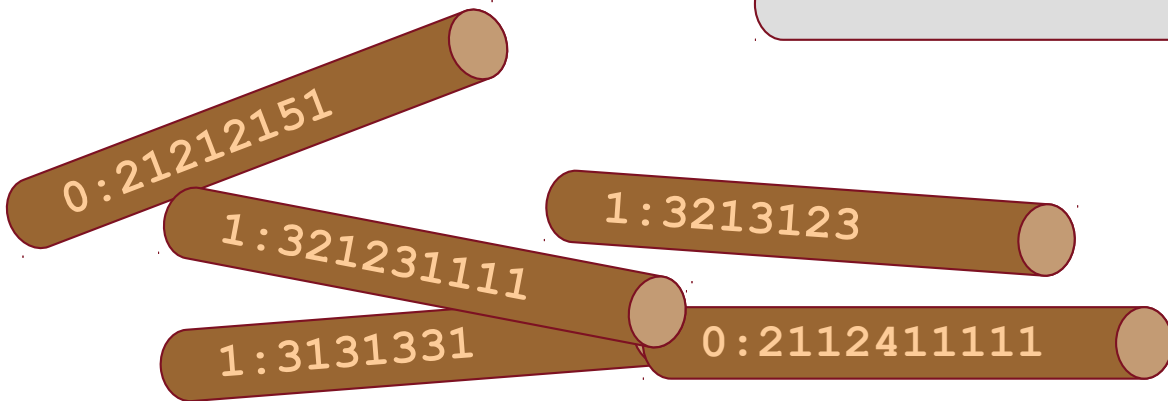
Beaver Ann uses a secret code to send messages to her friends. She is preparing a party and she wants her friends to give the entrance password.

She has engraved a secret party message for her friends on five sticks, but she got the order of the sticks mixed up.

Question

What order, from top to bottom, should the sticks be in?

1	1	1	0	0	1	0	0	1	1	1	0	1	0	1
0	0	1	0	1	1	0	0	0	0	1	0	1	0	1
1	1	1	0	0	1	0	0	0	1	0	0	1	1	1
0	0	1	0	0	1	0	0	1	0	0	0	0	0	1
1	1	1	0	1	1	1	0	1	1	1	0	0	0	1



Answer

The correct answer is:

1:321231111

0:2112411111

1:3213123

0:21212151

1:3131331

Each stick concisely describes a row of binary encoding the message. The first digit on each stick tells if the line of the image starts with white or blue square (0 for white and 1 for blue).

Numbers after the colon (:) indicate how many squares of each alternating colour will follow. By observation we may deduce that the stick starts with the first binary digit, then, after the colon (:), it lists the number of zeros or ones that follow.

For instance, 1:321231111 means that we have 3 ones, 2 zeros, 1 one, 2 zeros, 3 ones, and so on, corresponding to

the first row of the message 1 1 1 0 0 1 0 0 1 1 1 0 1 0 1.

The first line of the image begins with blue square (1). The colon (:) is followed by a digit indicating the number of squares in that colour (3 for three blue squares). This is followed by 2 white squares, followed by 1 blue, followed by 2 white, and so on.



Party Message - continued

Answer - continued

The top stick should have following code:

first blue square:	3 blue	2 white	1 blue	2 white	3 blue	1 white	1 blue	1 white	1 blue	1 white
1:	3	2	1	2	3	1	1	1	1	1

The second line of the image begins with white square so the first digit on the stick will be 0. There are 2 whites followed by 1 blue, followed by 1 white, followed by 2 blues and so on.

first blue square:	2 white	1 blue	1 white	2 blue	4 white	1 blue	1 white	1 blue	1 white	1 blue
0:	2	1	1	2	4	1	1	1	1	1

The following three sticks are matched to the corresponding lines of the image in the same way.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Pattern Recognition, Modelling and Simulation, Algorithms

Concepts: Data Collection, Data Representation, Data Interpretation, Specification, Algorithms, Digital Systems

The task introduces several data encoding concepts:

- First the password is represented as an image
- Then the image is encoded as a bitmap
- Finally, the bitmap is turned into the secret message using run-length encoding

Having several different ways to represent the same information and the need to convert from one representation to another is common in data processing.

The run-length encoding (RLE) introduced in this task is a very simple form of lossless data compression. The RLE algorithm exploits sequences having the same value occurring many consecutive times. In general, each such sequence is encoded by storing the full value only once, accompanied by the repetition count. Note that while in most cases the size of output is smaller than the size of the input, in the worst case it may actually be larger.

In this task storing of the values is also omitted, under the assumption that for each next count the corresponding value is the opposite of the previous one. (Can you think of an example image where this approach would get Ann in trouble? Can such images appear as representations of passwords?)

The task requires the student to use the skills of generalization and algorithmic thinking. First the student has to deduce the RLE algorithm by comparing the corresponding lines of the image and the message in the example. The student then has to apply this algorithm to the last line of the image to obtain the last line of the message.

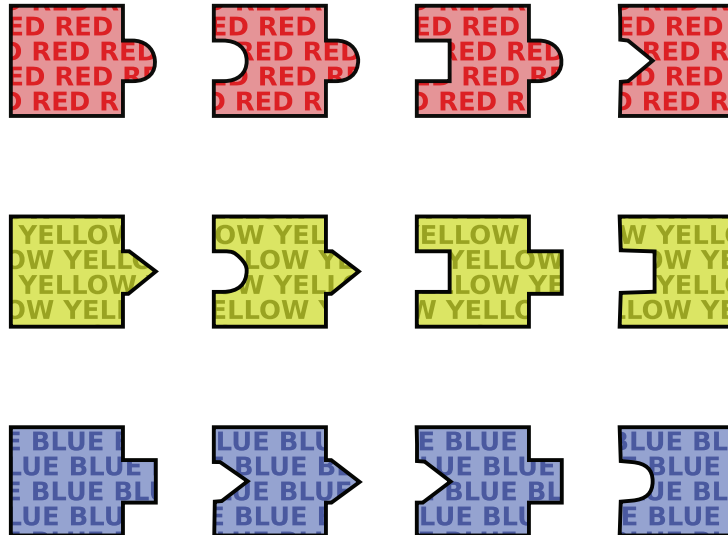
Restoring an image from the encoded result and then figuring out the characters or numbers it represents will not be too difficult for humans. However, the longer the sentence, the more time and effort it takes. To automate these tasks with a computer, people need to find out and tell the computer what to do and how. This is why students need to develop their Computational Thinking skills.



Jigsaw Puzzle

Beaver David has an unlimited amount of jigsaw puzzle pieces.

The pieces come in 12 different shapes, and 3 different colours: red, yellow and blue.



Using these pieces, he can create various colour sequences, for example:

BLUE → YELLOW → YELLOW → RED → YELLOW → BLUE → YELLOW → YELLOW



Each sequence must begin with a start piece (a piece with a flat left side) and finish with an end piece (a piece with a flat right side). Also, David cannot join two pieces on their flat sides.

Question

Which one of the following colour sequences can't be constructed using David's set of jigsaw pieces?

YELLOW → BLUE → BLUE → RED → RED → RED → BLUE

BLUE → YELLOW → RED → YELLOW → RED

RED → RED → YELLOW → BLUE → BLUE → BLUE

BLUE → RED → YELLOW → BLUE → RED → YELLOW → RED

Answer

The correct answer is C.

David can't join any of the blue pieces with the blue end piece. So, the sequence C. is impossible using David's jigsaw pieces.

Continued on next page



Jigsaw Puzzle - continued

It's Computational Thinking

Computational Thinking Skills: Pattern Recognition, Modelling and Simulation, Algorithms, Evaluation
Concepts: Specification, Algorithms

In this task you had to combine puzzle pieces in a certain order. Some of those combinations would fit together, some wouldn't. This is an important topic in Computer Science because very often, when different elements are combined together, there are rules in place that allow some combinations and reject others. Combinations that follow the rules are called well-formed.

In this task, the rules for these combinations were given by the shape of the pieces. In other cases it might become quite difficult to decide whether a given combination is well-formed or not.

Numbers are a practical example for such rules.

123, 13.2, 1000, 0 are all valid, well-formed numbers.

123\$45, 12 00, 13.2.5 are not.

Mathematical expressions can be well-formed:

$2+3$, $4*2$, $2/3$

Or they are not:

$2+*5$, $4*$, $/3$

The concept of well-formedness can be found throughout Computer Science, especially in programming.

More often than not computer systems are unforgiving when they are faced with malformed (not well-formed) elements but that's not only because the system wouldn't be able to cope with that element but to let the user know that he provided such an element which was probably unintentional.

In some cases computer systems silently accept malformed elements and try to make the best of the situation. This is done by web browsers because users prefer to see a webpage with a small flaw than no webpage at all.



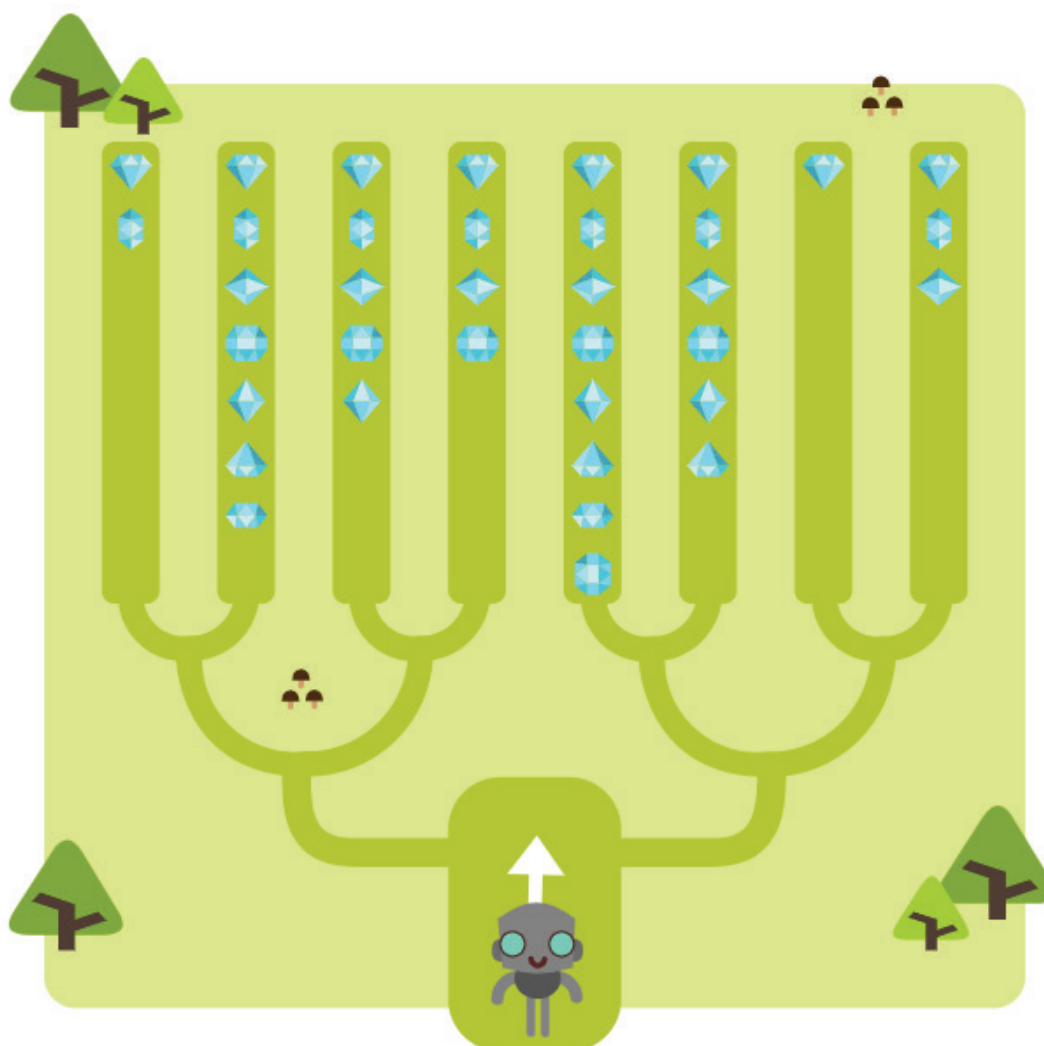
Robot Maze Game

Amoura and Banks have control of a robot in a maze with gems. In the picture below, the robot is shown at the start of the maze. The robot will follow the path until a fork in the maze is reached. One of the players decides which path (left or right) the robot should take. The robot will then follow the path again until another fork is reached, and so on. The game ends when the robot picks up some gems.

- Amoura and Banks take turns deciding.
- Amoura wants to make the robot pick up the highest number of gems possible.
- Banks wants to make the robot pick up the lowest number of gems possible.
- Amoura and Banks both know that they will try to outsmart each other.
- For example, if Banks sends the robot towards the fork with 3 or 1 gems, he knows that Amoura will send the robot right, to the 3 gems.
- Amoura takes the first turn.

Question

Click on the set of gems the robot will pick up if both players know that they will try to outsmart each other.



Answer

The correct answer is the path with the set of 5 gems.

Continued on next page



Robot Maze Game - continued

Answer - continued

Amoura knows that if she makes the robot go right on the first turn, Banks will not make the robot go left. Thus, the robot can never get 6 or 8 gems. If Amoura makes the robot go right, the robot can only get 3. It is possible for the robot to get more if it goes left. (We see later that it's not possible for the robot to get 2 gems if it goes left, so it can only get either 4, 5, or 7 gems which are all strictly bigger than 3.) Thus, on the first turn, Amoura makes the robot go left.

Banks knows that if he makes the robot go left on the second turn, Amoura will not make the robot go left to collect 2 gems. If Banks makes the robot go left, Amoura will make the robot collect 7 gems. It is possible for the robot to get fewer gems if it goes right (either 4 or 5 gems). Thus, on the second turn, Banks makes the robot go right.

Finally, on the third turn, Amoura will make the robot go left to collect 5 gems.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Modelling and Simulation, Algorithms, Evaluation

Concepts: Specification, Algorithms

When there are just three levels a logical reasoning suffices to solve the task. When there are more possibilities, game-playing AI use some kind of algorithm which expands the reasoning like the one above, to figure out which move to make. They work by first considering all the possible ways the game can play out after a few turns, give a score to each of these ways (for example, the number of pieces the AI will have left in a chess game), and make the move that maximizes the score.

The game-playing AI assumes that the player is smart and will try to make a move the minimizes the score (even if the player isn't completely aware of this). Hence, it will try to get the maximum of the minimum of the scores. That's why this algorithm is sometimes called a "maximin" search.

Or, reversing the roles (for example, scoring by the number of pieces the player will have left instead), the AI will try to get the minimum of the maximum. In other words, it will perform a "minimax" search. If the search goes on for several levels, the AI is actually computing the minimum of the maximum of the minimum of the maximum of the... But we just call it "minimax" for short.



Grocery Stores

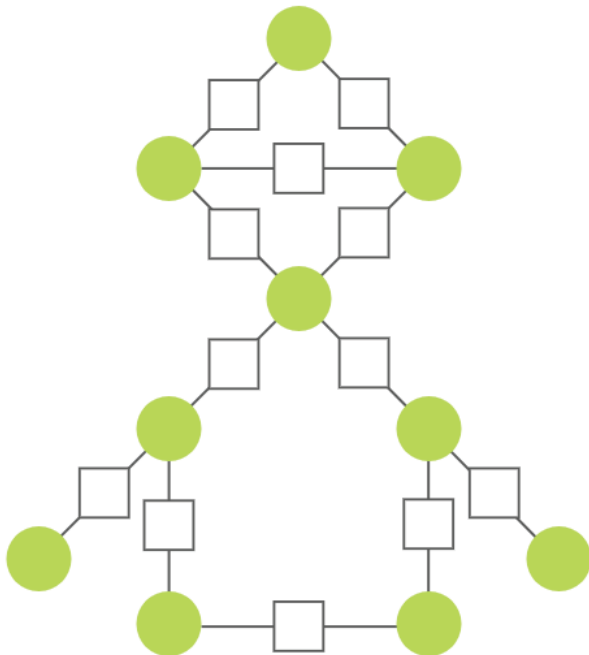
A town planner has decided that every town needs to have at least one grocery store close by. A grocery store is said to be “close by” a town if it can be reached without passing through another town to get to it.

In the diagram below:

- Green circles represent towns.
- Squares represent where grocery stores could be opened.
- The lines connecting them represent roads.

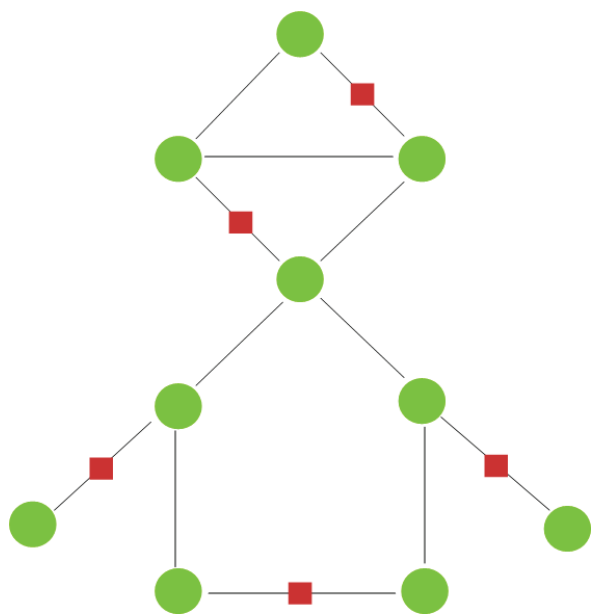
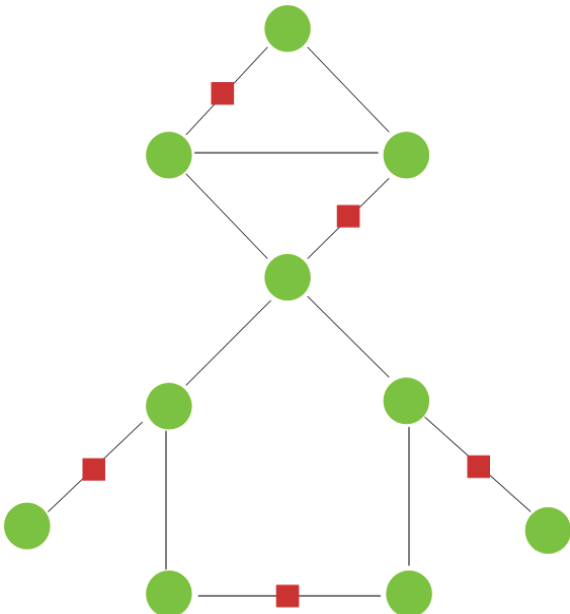
Question

Click on the squares in the map below to open the minimum number of grocery stores that will satisfy the town planner.



Answer

There are two possible solutions. In each case 5 grocery stores must stay open, as shown in the diagrams below:



Continued on next page



Grocery Stores - continued

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms
Concepts: Abstraction, Data Representation, Data Interpretation, Specification, Algorithms, Impacts

The problem in this task is known as the Minimum-Edge-Cover problem. The problem is to find the minimum amount of edges so that every vertex of the graph is 'covered by' (incident to) one of the edges selected. In this question, the villages are the vertices which need to be covered and the streets with the grocery stores are the edges, covering the vertices.

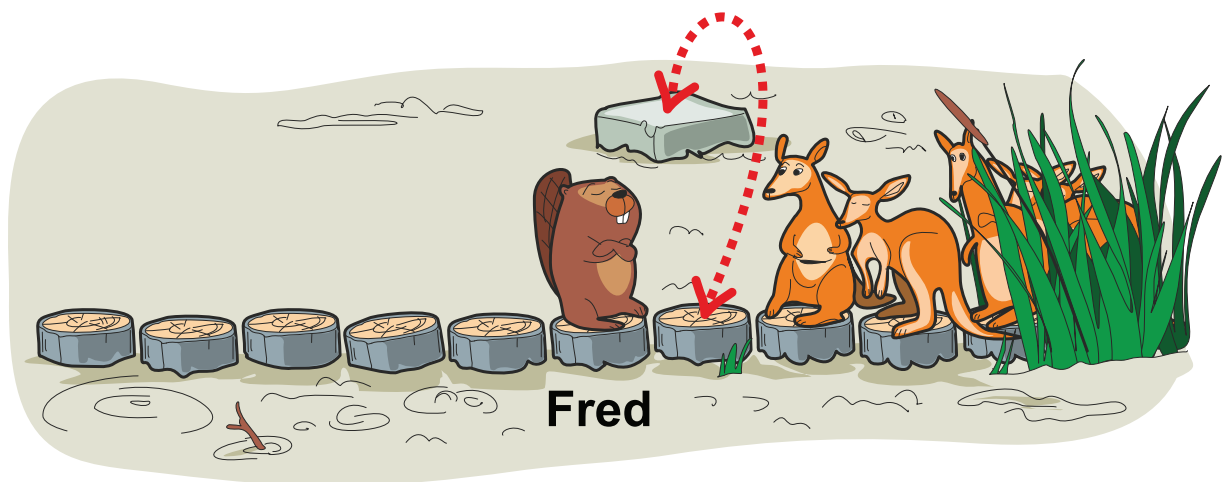


Beaver vs. Kangaroo

While crossing a swamp on a log path, Fred the beaver meets a group of kangaroos going in the opposite direction. Nobody wants to become wet or dirty so they stay on the path. The kangaroos realise that from one specific log it is possible to jump onto a stone next to the log path and then jump back again.

However, only one kangaroo can stand on the stone at a time. The kangaroos want to get past Fred and don't mind going back a few logs when they meet him.

Fred is stubborn and refuses to go back to the start of the log path and he won't take more than 10 steps backwards.



Question

If Fred acts this way, how many kangaroos can pass him?

More than 10 kangaroos

Exactly 10 kangaroos

Exactly 6 kangaroos

Exactly 5 kangaroos

Exactly 4 kangaroos

Fewer than 4 kangaroos

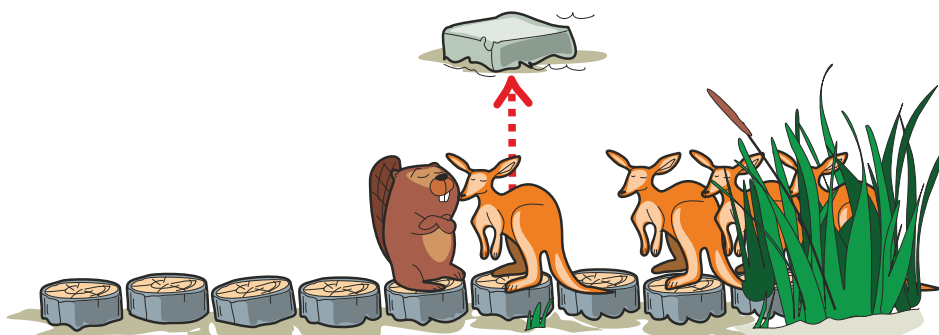
It is not possible to determine.

Answer

Correct answer is: Exactly 6 kangaroos can pass Fred.

All beavers except Fred can be ignored for now because they're willing to go back all the way. For Fred to let one kangaroo pass, this could happen.

1. The kangaroo jumps onto the stone.



Continued on next page



Beaver vs. Kangaroo - continued

Answer - continued

2. Fred goes two steps forward.

3. The kangaroo jumps back to the log path and can continue forward.

4. Fred goes back two steps to give another kangaroo the possibility to jump to the stone.

By executing this sequence of steps 5 times Fred can let 5 kangaroos pass by going back 10 steps in total; then one more kangaroo can pass because Fred will then be in its initial position again.

So a total of 6 kangaroos can pass if Fred takes a step back 10 times.

This can be expressed by a mathematical formula. If Fred wants to let k kangaroos pass, he'll have to walk $s = 2 \times (k - 1)$ steps back.

Solved for the number of kangaroos k , the formula is $k = 0.5 \times s + 1$.

It's Computational Thinking

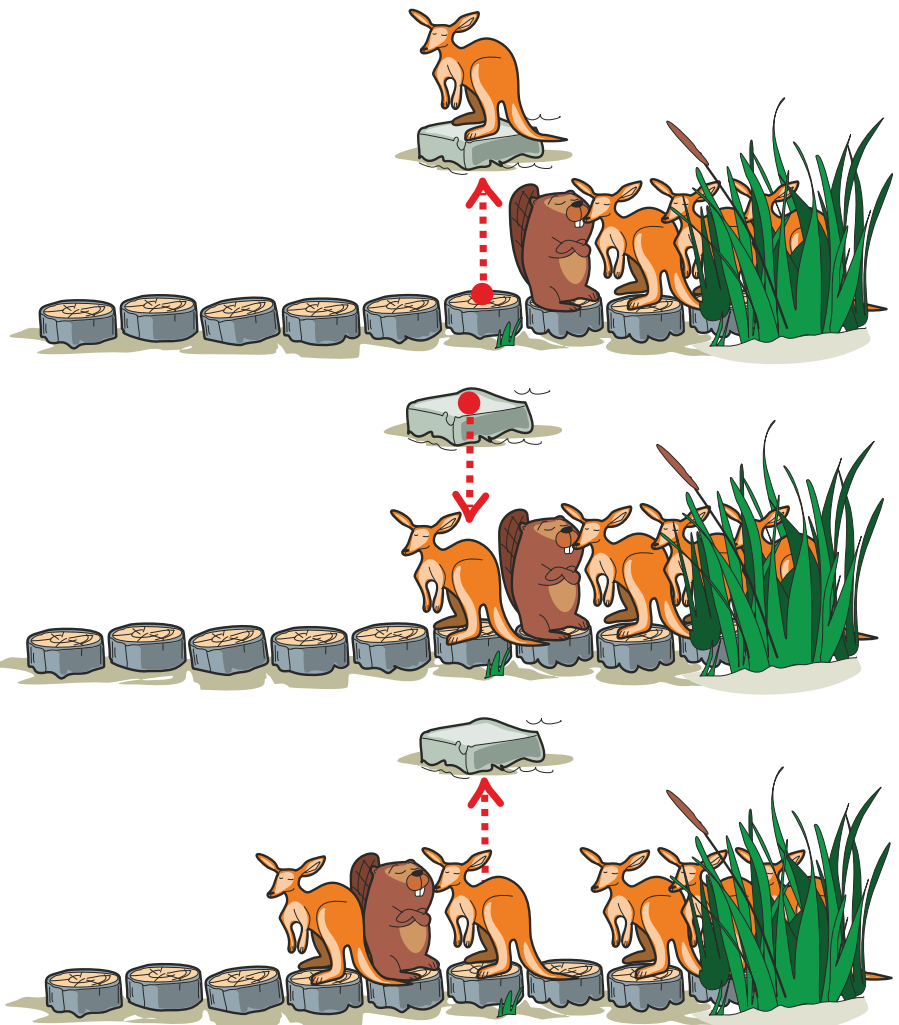
Computational Thinking Skills: Decomposition, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Specification, Algorithms

Algorithms are essential to the way computers process data and complete task steps in a specific order:

- Changing the content of a variable: Each log and the stone is a place where information can be stored like in a variable and the beavers and the kangaroos is the data to be stored in these places.
- Ordering of the steps: There must be some arrangement to make kangaroos pass the beavers by moving them to different places. Because they cannot pass directly, creating a schedule for what to do when helps to solve the problem.
- Repeating necessary steps as much as needed: In this case the same sequence of movements is repeated several times, which also is a typical concept of Computational Thinking: solve a small problem once and repeat the solution as often as necessary.

Thus, an algorithm can be considered to be any sequence of operations including repetitions that can be simulated by computer systems. Recognition of patterns in algorithms (similar steps that are repeated) can be turned into reusable code for a quick and automatic solution of a problem (as the formulation here). The logs and the stone are like registers in a computer processor or on a tape drive that can store data.





Hansel and Gretel

Hansel and Gretel are playing a game. There are 3 black stones and 7 white stones. On each turn, a player may take 1 or 2 black stones OR 1, 2, or 3 white stones. The player who takes the last stone(s) of any colour wins the game.

Question

Gretel goes first. Which stone(s) does she need to take in her first turn to ensure she wins the game?



1 white stone

2 black stones

3 white stones

It doesn't matter how many stones.

Answer

The correct answer is (C).

For the 3 black stones, if Gretel removes 1 or 2 stones, then Hansel can remove the remaining stones (i.e. if Gretel removes 1 black stone, Hansel can remove 2 stones, and vice versa). This makes Hansel the last player to remove a stone(s). So, the winning strategy in black is to leave 3 stones.

For the white stones, a player can only remove 1, 2, or 3 stones. With any group of 4 white stones, the second player will always be able to remove the remaining stones (i.e. if 1 white stone is removed, the other player can remove 3 stones, and so on). The winning strategy in white is to leave 4 stones.

For Gretel to win, she must leave 3 black stones and 4 white stones. This is only possible by removing 3 white stones in the first move. If Gretel fails to apply the strategy in her first move, Hansel can steal the strategy, so other options will not be a winning move for Gretel.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms

Concepts: Abstraction, Data Representation, Data Interpretation, Specification, Algorithms

Games are an important topic of study for Computer Science. They can be used to model many real-life interactions. An important algorithmic question in the study of games is to find a winning strategy for one of the players.

In this case, we can work backwards from positions where Gretel can win with a final move and classify positions as winning for Gretel and losing for Hansel. From this, we can derive a property that Gretel must maintain to always stay in a winning position. This is called an invariant, a property that she must maintain with every move made and which Hansel cannot destroy with any move that he makes. If she can maintain this invariant, she has a winning strategy.

This kind of analysis can be extended to complicated games like chess and Go, but then the number of possible positions becomes extensive, and we need to use other approaches like AI to “learn” a good strategy.

Nim was one of the first games ever implemented on electronic device. And maybe that combinatorial game theory has solved Nim mathematically for any number of initial heaps and objects, and there is an easily calculated way to determine which player will win and what winning moves are open to that player. So, the strategy for computer-bot could be directly programmed, unlike, for example, chess.

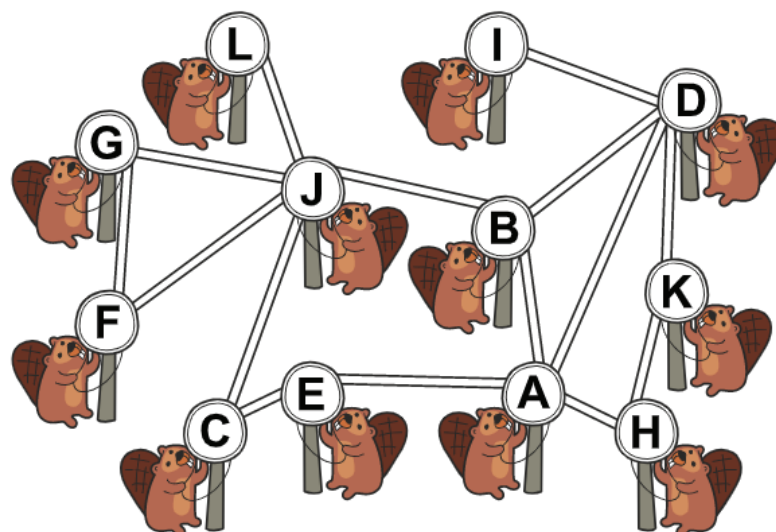


News Travels Fast

The 12 beavers in this colony want to be informed about news events as quickly as possible. Each beaver has its own pole that is connected with ropes to the poles of other beavers. They use the poles and ropes to send each other messages. Whenever a beaver hears some news, he immediately uses all the ropes that connect to his pole to inform the other beavers.

Example

If the beaver at pole F hears some news, she informs the beavers at poles G and J. The next beavers to hear the story will be the beavers at poles L, B and C, and so on, until all the beavers know the latest news.



Question

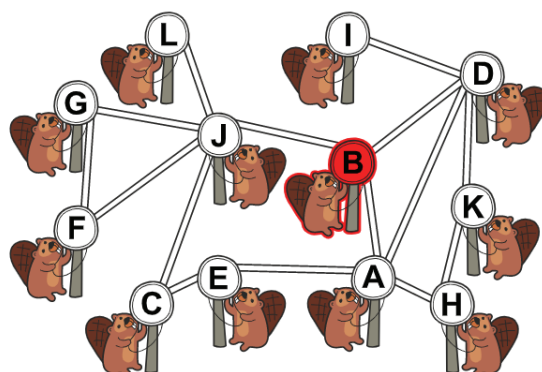
Click on the letter of the beaver that you should inform if you want your news to reach all the beavers as soon as possible.

Answer

The correct answer is Beaver B.

If the story is told to Beaver B, it will be distributed to all the other beavers in 2 steps. In the first step Beaver D, Beaver A, and Beaver J are informed.

They then inform all the other beavers. So all the poles are at a distance of at most 2 steps from Beaver B (the initial caller). There is no other pole for which this is possible, or even a pole with a distance of 1 step to all other poles.



It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Collection, Specification, Algorithms, Digital Systems, Interactions, Impacts

Many real-life problems are solved by computer programs after representing them as a graph. A graph consists of a set of vertices or nodes (usually depicted as points) and a set of edges (usually depicted as line segments, possibly curved), that connect these vertices. In this task, the underlying problem is to find the so-called graph centre.

The centre (or Jordan centre) of a graph is the set of all vertices of minimum eccentricity, that is, the set of all vertices u where the greatest distance $d(u, v)$ to other vertices v is minimal. Here, there was only one such vertex – city 2.

Finding the centre of a graph is useful in facility location problems, where the goal is to minimize the worst-case distance to the facility. For example, placing a hospital at a central point reduces the longest distance the ambulance has to travel.

If we have a small graph, like in our task, we can use trial and error, as it is simple to calculate the distance between all pairs of vertices. For large problems the centre can be found using with more sophisticated algorithms like the Floyd–Warshall algorithm.

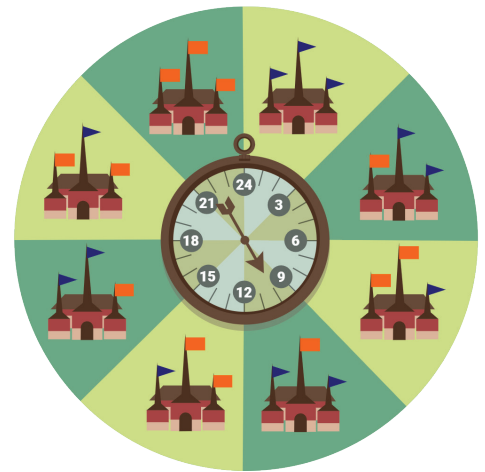


Beaver Time

Beavers divide the day into 8 intervals of 3 hours and because they have a relaxed attitude towards time, they only want to know the current interval.

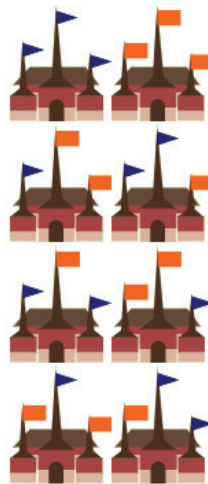
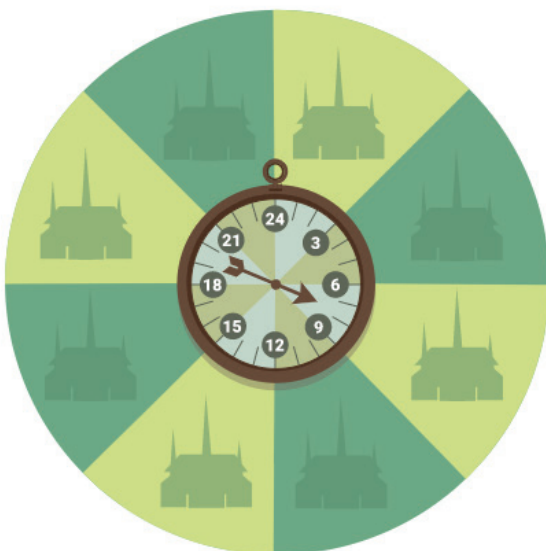
To tell the time they look at the town hall, a building with three steeples. A flag flies on top of each steeple, either a blue triangle or an orange square. Each pattern signals a time interval as shown on the right.

The mayor is proud of the system because only one flag changes from each interval to the next, except at midnight when all three flags need to be changed.



Question

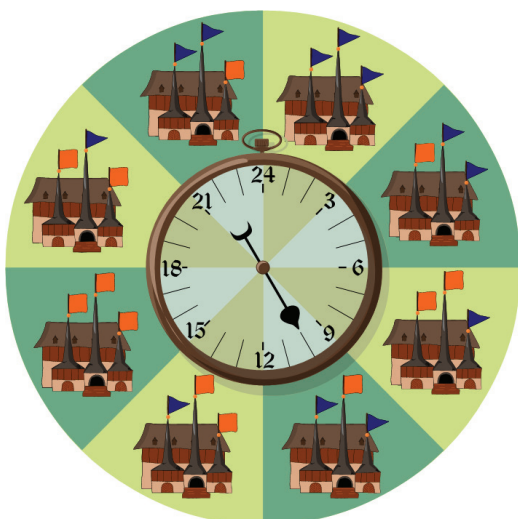
Create an even more efficient flag pattern that never requires changing more than one flag at a time.



Answer

The following is one of many possible solutions.

Let's use a binary representation of flag states – 0 for blue triangle and 1 for red square.



Then we have 8 variations: 000, 001, 010, 011, 100, 101, 110, 111. We need to put them in the circle where any two adjacent numbers differ only by one place. We start from any two numbers with such feature: 000 and 100. Then we add 110 (or 101) to the right and get 000, 100, 110. Now we can add 010 and get system of 000, 100, 110 and 010. The only possible next number is 011. Then we add 111 and get 000, 100, 110, 010, 011, 111. There are two remaining numbers: 001 and 101. The second one is similar to 111, except for one digit, and the first one is similar to 000, except for one digit. Thus, we get 000, 100, 110, 010, 011, 111, 101, 001. This system is perfect.

As we had options to choose the first pair and then the third number, there are many possible systems.

Continued on next page



Beaver Time - continued

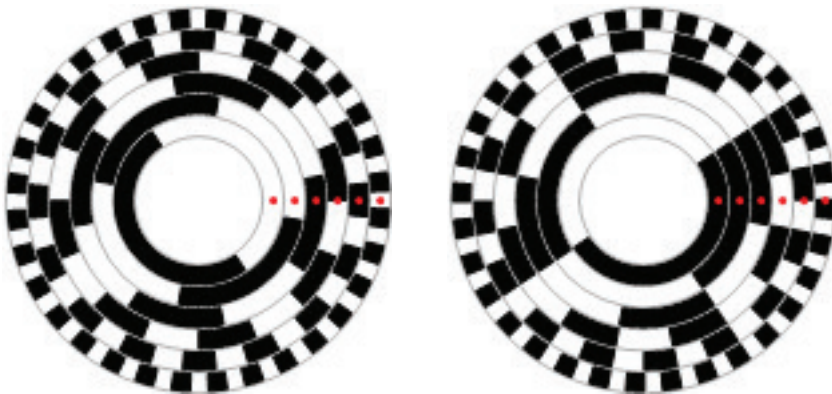
It's Computational Thinking

Computational Thinking Skills: Modelling and Simulation, Algorithms, Evaluation

Concepts: Data Collection, Data Representation, Data Interpretation, Specification, Algorithms

Such a pattern is called a Gray Code and it has a lot of applications. The fact that only one bit changes between consecutive code words can for example lead to less energy consumption. Flipping more than one bit necessarily uses more energy and the normal binary code flips all bits at once when wrapping around.

A famous application of Gray codes in engineering is measuring the angle of a rotating disk. We draw a Gray code on the disk, as shown in the picture in the left. Installing an array of light sensors that can distinguish the colours black and white (shown in red) in a fixed position allows us to read off a binary number encoding the current angle of the disk rotating below the sensors.



In the left picture, we can see how the light sensors read the code word 001010 or 001110 because the fourth light sensor is right at the edge between a white and a black bar. Either way, there is no harm done because the angle is right between the angles encoded by those two code words. Using a different code, such a situation could be disastrous, however. For example, in a normal binary code, as the one seen in the right picture, the code words 111010 follows the code word 111001. If the disk were at the edge between the corresponding angles the sensors could read the code word 111011, which encodes an angle that is quite a bit off. In the worst case, the sensors would be positioned over the border between the all-white code word 000000 and the all-black code word 111111. In this case, all sensors can switch between 0 and 1 at any time and leading to an arbitrary angle measurement.



New Neighbour

A new neighbour has moved into a new house in Beaver Village. The village has a rule to decide on the colour of a new house:

The colour of a new house should be the same colour as the majority of the k nearest houses. If there is a tie, then use $k+1$ instead of k .

The number k is unknown to us. The map of the Beaver Village is shown below.



Question

According to the rule the new house must be painted orange.
What is the minimum possible value of k ?

1

2

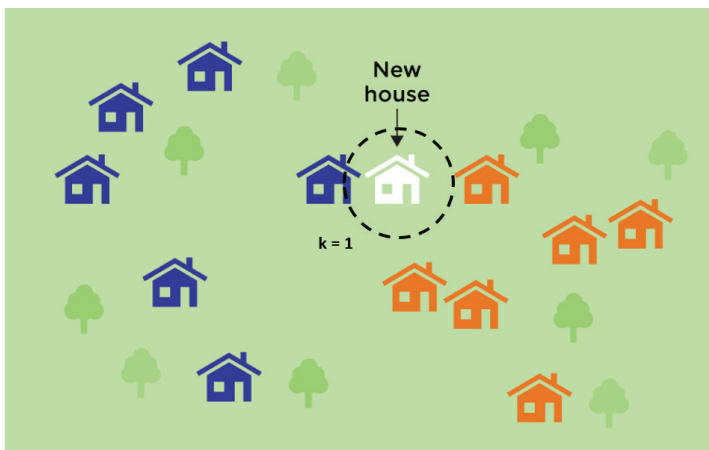
3

4

Answer

The correct answer is C, $k=3$.

You can find the answer by increasing k , beginning from 1 and for each k checking if the correct colour is selected by the rules. First let's say $k=1$. Draw a circle centred at the new house to include only 1 other house, as shown below. There is only one blue house in the circle.



According to the rule, this must mean that the new house should be coloured blue, however we are told that the new house must be orange. This means we must increase the value of k from 1 to 2.

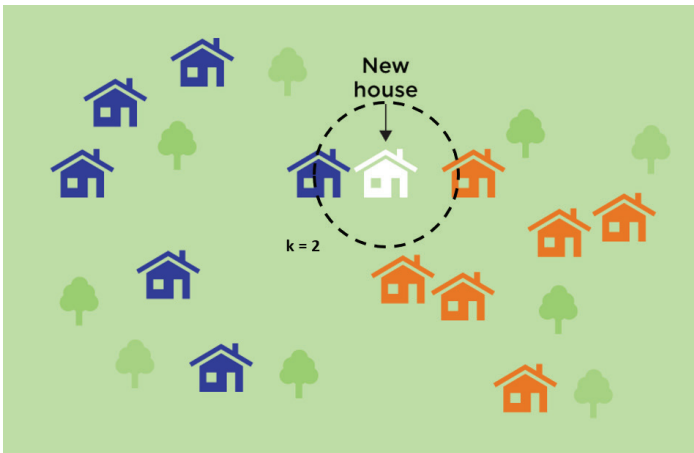
Let's say $k=2$ now. Draw a circle centred at the new house to include exactly two houses, as shown below. We can see that there is one blue house and one orange house in the circle. In this case, there is a tie, and so the colour for the new house cannot be determined, so k must be increased again.

Continued on next page



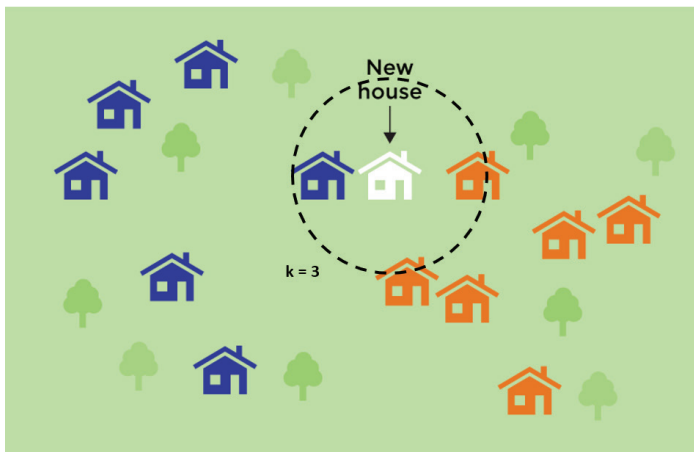
New Neighbour - continued

Answer - continued



We now have to check the situation for $k=3$. Draw a circle centred at the new house to include exactly 3 other houses, as shown below.

There are two orange houses and one blue house within the circle. Thus, the majority is orange, and the colour for the new house must be orange. This was the first time that the result gave us orange for the new house, so the minimum value for k is 3.



It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms

Concepts: Abstraction, Data Collection, Specification, Algorithms

This task involves key ideas regarding the k Nearest Neighbours (k -NN) algorithm, which is used for machine learning in data classification applications. E.g. to classify pictures of blossoms or economic development of countries. k -NN algorithm tries to classify data points by comparing each data point with the nearest k other data points and adjusting the value of k to fit the overall structure of the data. If the value of k is too small or too large, the classification may become useless.



Air Conditioning

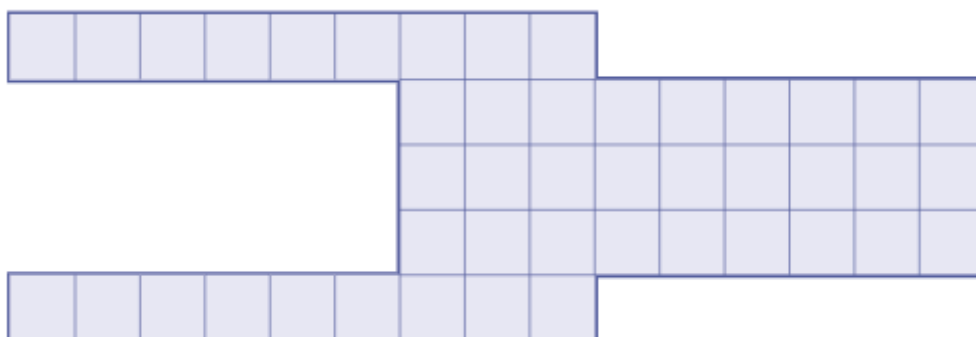
A beaver has discovered that it is more efficient to use air conditioning to heat her home than central heating. She has therefore decided to install 4 air conditioning units into the house shown below.

The house is formed of cells. An air conditioning unit occupies one cell and heats (or cools) that cell instantly. The hot air from a cell takes 1 minute to spread to all its neighbouring cells as shown on the right. The image shows how it takes 3 minutes to warm a small room with one air conditioning unit placed on the cell marked 0.

2	2	2	2	2	3
2	1	1	1	2	3
2	1	0			

Question

Place 4 air conditioning units in the house below, so that the entire house is heated in as few minutes as possible.



Answer

The correct answer is B, 2min.

If the heater units are placed as in the image, it will take 2 minutes to reach each cell.

2	1	0	1	2	2	2	2	2				
									1	1	1	2
									1	0	1	2
									1	1	1	2
2	1	0	1	2	2	2	2	2				

It is not possible to heat every cell in 1 minute. To fill the entire house in a minute, all the cells would have to be in the immediate vicinity of a heater unit. The house has 36 cells, thus each unit must be connected to a different set of 9 cells (including itself), but given the shape of the house this is not possible. Alternatively, we can see that the

house has two paths of length 15. Since each unit can cover a maximum of 3 cells in length, it would be impossible for the 4 units to cover even a single path. Thus, the minimum time is 2 and the correct answer is B.

It's Computational Thinking

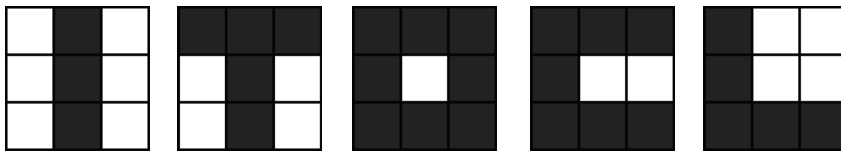
Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms
Concepts: Abstraction, Data Collection, Data Representation, Specification, Algorithms, Impacts

This task is related to the propagation in a graph, as a model of the room. In this case, each cell in the room is represented by a node and two nodes are connected by an edge if the corresponding cells are neighbours, as defined in the task body. In this task, we have to select special nodes (cells) such that each node in the graph (cell in the room) can be reached (by heat air) along a smallest number of edges (neighbouring cells).








Heat Maps

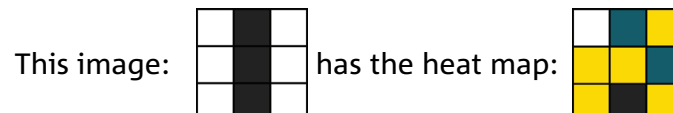
A letter machine can recognise these five images, which represent the letters I, T, O, C and L.



The letter machine uses heat maps in the recognition process. In the heat map of an image, the colour of a square indicates the uniqueness of the pixel colour at this position in the other letter images. The lighter the colour, the more unique the pixel is.

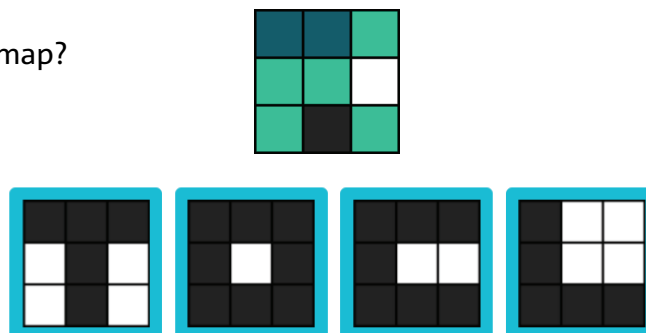
-  **Unique:** None of the other images has the same pixel colour at this position.
-  **Rather unique:** Only one of the other images has the same pixel colour at this position.
-  **Not unique:** Two of the other images have the same pixel colour at this position.
-  **Rather common:** Three of the other images have the same pixel colour at this position.
-  **Common:** All other images have the same pixel colour at this position.

Example



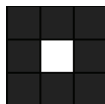
Question

Which image has this heat map?



Answer

The correct image is:



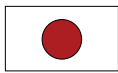
The given heat map is white at position 3 in the second row. That means that the corresponding pixel of the image must have a unique colour. The answer above is the only image with a black pixel at this position. Thus, it is the only image with a unique pixel colour at this position.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Pattern Recognition, Abstraction, Modelling and Simulation, Algorithms

Concepts: Abstraction, Data Representation, Data Interpretation, Specification, Algorithms, Digital Systems

A heat map is a graphical representation of data where the individual values contained in a matrix are represented as colours. You probably know heat maps in weather reports, indicating temperatures in different regions. Heat maps like in this Bebras task are used in image recognition. They assign a “perceptual importance” to each pixel and help to focus the attention of a deep learning network to certain areas of an image.



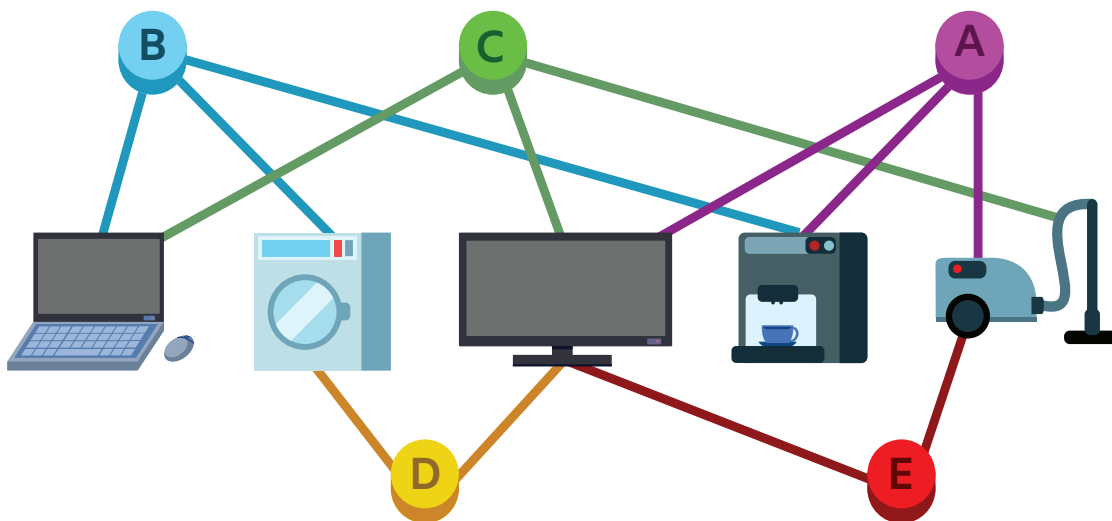
Household Appliances

In B-taro's house, there are five household appliances (computer, washing machine, TV, coffee machine, and vacuum cleaner), and five buttons (A, B, C, D, and E) to control those appliances. You can change the on/off state of the appliances by pressing the buttons. However, the buttons are designed to be inconvenient. As the buttons are connected to multiple appliances, each button changes the on/off state of multiple appliances at the same time.

- Button A is connected to TV, coffee machine, and vacuum cleaner.
- Button B is connected to PC, washing machine, and coffee machine.
- Button C is connected to PC, TV, and vacuum cleaner.
- Button D is connected to washing machine and TV.
- Button E is connected to TV and vacuum cleaner.

Question

Turn only the TV and coffee machine on!



Answer

The buttons that have to be pressed to turn only the TV and coffee machine on are: B + D + C + E.

In order to find the combinations of buttons leading to a given state of the appliances, we can observe that certain combinations of buttons can control a particular appliance. First, we can observe the simple combinations:

- A + E controls the coffee machine
- C + E controls the computer

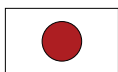
Then, we can observe that the washing machine is controlled by pressing B and then turning off the computer and the coffee machine, both of which we now know how to control. Hence, the washing machine is controlled by $B + A + E + C + E = A + B + C$ (since $E + E$ cancels).

This way, we can obtain the full list of combinations to control each device:

- Computer: C + E
- Coffee machine: A + E
- Washing machine: A + B + C
- TV: A + B + C + D
- Vacuum cleaner: A + B + C + D + E

So, to get the TV and the coffee machine turned on, we need to press their respective combinations, A + B + C + D + A + E, or just B + C + D + E, since A + A cancels.

Continued on next page



This question comes from
Japan

Years 3+4
Years 5+6
Years 7+8 Hard
Years 9+10
Years 11+12



Household Appliances - continued

It's Computational Thinking

Computational Thinking Skills: Decomposition, Modelling and Simulation, Algorithms, Evaluation

Concepts: Data Collection, Specification, Algorithms, Digital Systems

The system of appliances and buttons and changes of on/off states can be modelled by what is called a Finite State Machine (FSM). An FSM is used in Computer Science to model systems by representing all possible states and all transitions from one state to another.

For this task we can use a Finite State Machine to represent all possible on/off states of the set of electrical appliances and the transitions caused by pressing the different buttons.

The FSM for our task has an Initial State (S_0) where all appliances are in the off state. Then, when button A is pressed, we go to a different state, S_1 , where the computer and the washing machine are off, while the TV, the coffee machine and the vacuum cleaner are on. From S_1 , if we press A again, we go back to S_0 , with all appliances off.

However, if we press for example button E, we go to a state S_2 , where all appliances except the coffee machine are off, as shown below. Hence, we obtained the method to control the coffee machine: by using buttons A + E, as explained in the answer section.

Similarly, by developing all the transitions of the FSM, we could find the combinations that control each appliance.

Bebras Challenge 2021 Round 1

Years 9+10



Winners or Losers

Beavers are having a chess tournament. They have already played some games. In the picture below, arrows have been drawn from the winner to the loser.

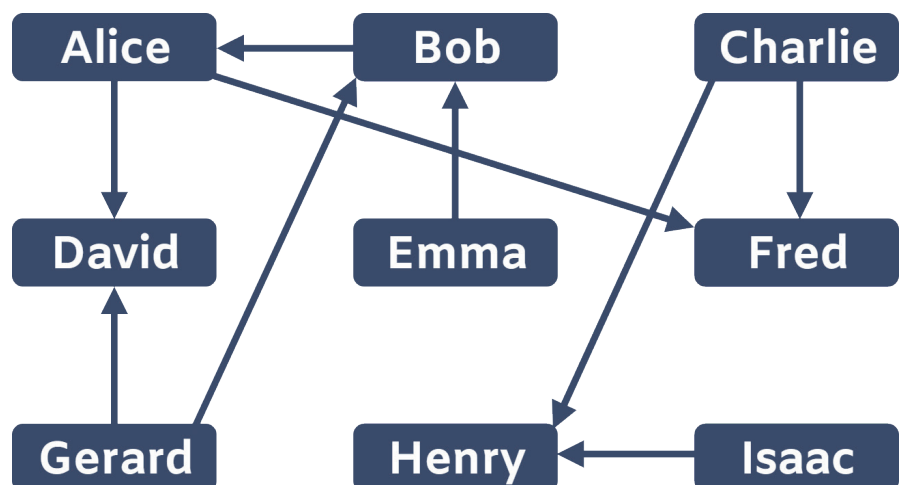
Example

Bob beat Alice and Alice beat David.

A special trophy is awarded to any beaver that beats every other beaver. Each pair of beavers will play each other one time.

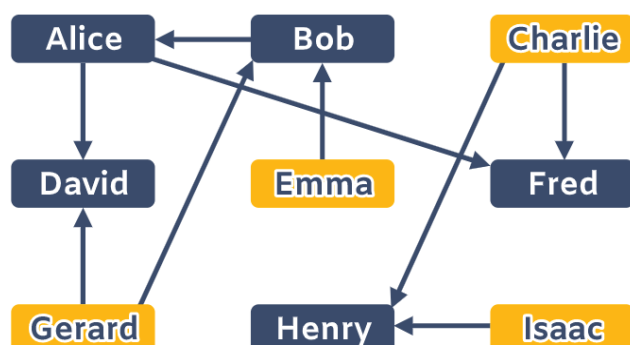
Question

Select the names of all of the beavers who still have a chance of winning the trophy.



Answer

Given the state of the competition, only Charlie, Emma, Gerard and Isaac are still able to win the special trophy. Every other beaver in the competition has already lost a match to another beaver.



It's Computational Thinking

Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms

Concepts: Abstraction, Data Interpretation, Specification, Algorithms

A directed graph (or digraph) is a way to represent (non-symmetrical) relationships between pairs of objects: objects are depicted as vertices, and object A is connected to object B by an arrow when A is in relationship with B.

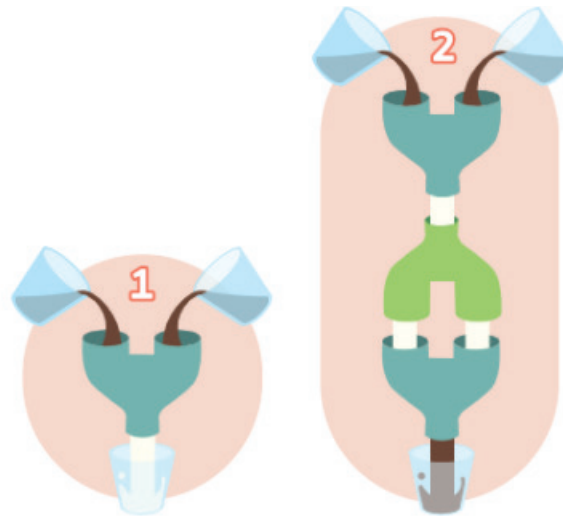
In this task the relationship is “winner”: some relationships come from actually played matches, others from the rule given in the text that makes winning transitive, a property normally not true for sports: indeed matches are exciting to follow exactly because if Bob wins against Alice and Alice wins against David, nothing guarantees that Bob would beat David in a real match.

The problem in the task, however, is common in informatics: some objects are not directly compared with each other and still we need to build an ordering where every pair of objects are compared: of course there can be more than one of such orderings. An example is when you have activities or programs that depend on others: to put on shoes you should already have put on socks, and to put on a coat you should already have put on a shirt but you can certainly find several different ways to properly order your dressing actions!



Magic Drink Machine

Beaver Cobi has a mysterious blue coloured machine as shown in Fig. 1. There are two funnels in the machine.



- If a beaver pours chocolate milk into both funnels, white milk comes out.
- If a beaver pours white milk into either of the funnels, chocolate milk comes out.
- If a beaver connects two machines and pours chocolate milk into both funnels as shown in Fig. 2, chocolate milk comes out at the end.

Note: the middle green connection has no effect on the type of milk.

Question

What type of milk should be poured into both funnels so that white milk comes out when connecting three machines?

Answer

To get white milk at the end, chocolate milk must be added to both funnels of the third machine.

To achieve that, chocolate milk must come out of the two machines above. To get chocolate milk from the two machines above, white milk needs to be added to one or more of the inputs for of both funnels for each machine. Since for these upper machines the inputs are the same in each funnel, white milk will need to be poured into each.

Therefore, in order to make white milk come out at the end, the milk to be poured on top is <white milk, white milk>.

It's Computational Thinking

Computational Thinking Skills: Modelling and Evaluation, Algorithms, Evaluation

Concepts: Data Interpretation, Specification, Algorithms, Digital Systems

An electronic circuit that receives one or more logic values and performs logical operations and outputs one logic value is called a logical circuit. Logical operations include AND, OR, NOT, and XOR, among which the definition of NAND is as follows.

Continued on next page



Magic Drink Machine - continued

It's Computational Thinking - continued

- $A \text{ NAND } B = \text{NOT } (A \text{ AND } B)$

The interesting thing is that you can express all logical operations with only one type of NAND.

- $\text{NOT } A = A \text{ NAND } A$
- $A \text{ AND } B = (A \text{ NAND } B) \text{ NAND } (A \text{ NAND } B) \rightarrow \text{Fig. 2}$
- $A \text{ OR } B = (A \text{ NAND } A) \text{ NAND } (B \text{ NAND } B) \rightarrow \text{Fig. 3}$

In the question, white milk represents 0, chocolate milk represents 1, and the milk machine itself performs a NAND operation. Figure 1 demonstrates a simple NAND operation, Figure 2 shows implementation of the AND operation using only NAND operations, and Figure 3 shows the implementation of an OR operation using only NAND operations.



Damaged Secret Table

The secret writing here is based on encoding the letters of the Latin alphabet with new symbols. The symbols are described in the following table. Unfortunately parts of the table have been wiped out and some parts are missing:



Question

Find the original plain-text of this cipher-text, even though the table is damaged.



INFORMATION SECRET

INFORMATICS IS COOL

MATHEMATICS IS COOL

INFORMATION IS COOL

Answer

The full encryption-table is:

The plain-text is: **INFORMATICS IS COOL**

To solve the problem one has to recognize that the new symbols are created by combining the labels of the rows and columns. The label of the row is the lower part of the symbol and the label of the column is the upper part of the symbol.

	I	II	III	△	△	X	X
□	A	B	C	D	E	F	G
◐	H	I	J	K	L	M	N
▣	O	P	Q	R	S	T	U
▽	V	W	X	Y	Z		

Continued on next page



Damaged Secret Table - continued

Answer - continued

The missing symbols of the Latin alphabet can be easily inserted. One only needs to know the order of the letters of the Latin alphabet.

When looking at the rows, only the label for the first row is missing. Knowing that the rows indicate the lower part of the symbol, by looking at the cipher-text you can recognize which one is missing.

For some Latin characters you already know the secret symbol. By inserting these into the cipher-text you already obtain a part of the plain-text. By trial and error one can fill in the remaining blanks.

An alternative way to solve the problem is without rebuilding the table: It is sufficient to note that the first symbol corresponds to letter I which excludes answer C. The eleventh symbol corresponds to letter S, but a different letter is in this position in options A and D. So the only possible answer out of the four remains answer B.

It's Computational Thinking

Computational Thinking Skills: Pattern Recognition, Algorithms, Evaluation

Concepts: Data Representation, Data Interpretation, Specification, Algorithms

Protecting data is a task that is older than 4000 years. Cryptography is a crucial part of informatics that focuses on developing secret codes in order to protect confidential data or confidential messages.

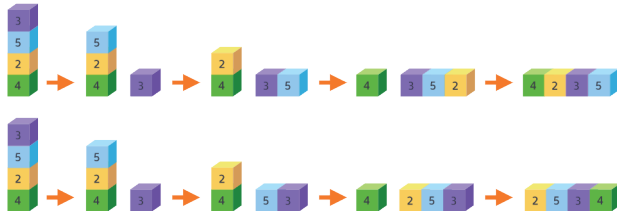
In older cultures the main way of developing secret codes was creating a simple method for decryption that can be easily memorised. The secret code here was invented for the Bebras challenge but is exactly that kind of code.

It is a simple substitution cipher in which there is a one-to-one connection between the set of letters of the alphabet and the set of symbols chosen to encode the letters.



Creating Numbers

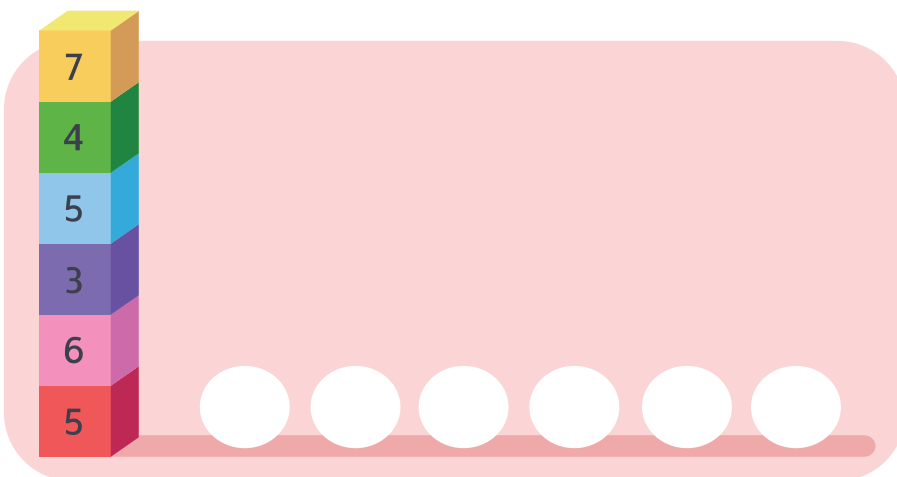
Olivia the beaver, is playing with blocks. Each block has a single digit on it. She loves to make a big tower and then use the blocks one by one, from the top, to form a number. Each time she takes a block off the tower, she can place it to the right or to the left of the number she is forming. The following figures show a tower of 4 blocks and two possible numbers that can be formed from it (4235 and 2534):



Olivia just built a new tower of 6 blocks and she wants to create the smallest possible number from it.

Question

Can you help her by dragging the blocks into their correct position?



Answer

The correct answer is 347565.

The smallest possible number should start with the smallest digit in the tower, which is 3. All the numbers that are below 3 in the tower cannot be placed to the left of it, so will need to be added to the right of the number being formed, so the final number ends with 65. Above 3 we have a smaller tower 547, from which we want to form the smallest possible number to come between 3 and 65. We can apply the same reasoning: the number should start with the smallest digit, 4, and 5 should be added to the right, so we get the number 475. This gives us the final answer, 475, preceded by 3, followed by 65.

It's Computational Thinking

Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Collection, Specification, Algorithms

Searching for the best solution (such as the minimum number) among all possibilities is a very common computational task known as optimization. We often use additional techniques to narrow down the possibilities and speed up the search. In this case, we use a greedy strategy to fix the smallest digit as the leftmost digit. We then use a divide-and-conquer approach to solve a smaller instance of the same problem for the blocks above the smallest digit.



Echo Cipher

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Do you know how to add two letters? It's easy: both letters become the numbers of their order in alphabet. The result will be a number, which we can turn into a letter by looking at which letter is in that position in the alphabet.

For example, $A+A=B$, $A+B=C$, $C+E=H$

If the resulting number is bigger than the number of letters in the alphabet, we can start again from the beginning of the alphabet, e.g. $Z+A=A$, $Y+C=B$

Lenka uses this addition of letters to encrypt her journal.

She works like this:

1. First she thinks of a number, which she calls shift.
2. She writes one word and then writes the same word again below the first one, but the second word is shifted to the right by a number of positions equal to the shift.

For example, if the shift is 1, the encryption looks like this:

	T	E	A	
+		T	E	A
1	T	Y	F	A

$E+T=Y$
 $A+E=F$

Question

Lenka encrypted the word COMPUTER, and came up with the word COMSJGUMTER. Which number is the shift?

1

2

3

4

Answer

The correct answer is 3.

We can tell that this is the correct answer with two pieces of information:

1. The encrypted word is 3 letters longer than the original word.
2. The first three letters are the same in both the original and encrypted word.

	C	O	M	P	U	T	E	R			
+				C	O	M	P	U	T	E	R
	C	O	S	M	J	G	U	M	T	E	R

Continued on next page



Echo Cipher - continued

It's Computational Thinking

Computational Thinking Skills: Pattern Recognition, Abstraction, Evaluation

Concepts: Abstraction, Data Representation, Data Interpretation, Specification, Digital Systems

We do not always want messages sent over networks to be easily read if someone intercepts them. These messages may contain passwords and other private information. Therefore, messages are encrypted, which turns them into secret messages. For this to work, the recipient of the message must be able to decrypt the secret message and reveal its original text. However, it should not be possible for unauthorized persons who capture a secret message to be able to translate it.

The encryption and decryption of secret messages is done using ciphers. There are many different types of ciphers. The cipher used in this task replaces each letter in the plaintext by a letter some number of positions down the alphabet. This number is given by the positions of individual letters of the same plaintext in the alphabet and it is not fixed. This cipher is easy to break for computers when the text is long enough because it does not change a number of the letters used in the language.

The science dealing with encryption is called cryptography. Modern cryptography is a large area of research that involves many other ciphers based on complex mathematics.

Sierpinski Triangle

To form a Sierpinski triangle pattern, a large white equilateral triangle has increasingly smaller black triangles removed from it. For each iteration, the following set of steps is repeated for all of the largest white triangles remaining in the pattern:

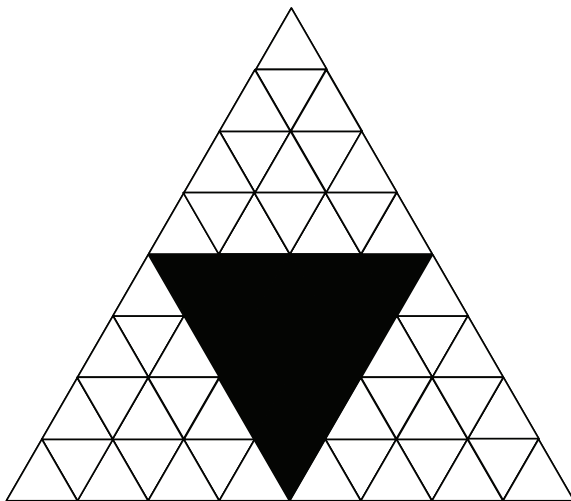
- Mark the centre of each side of the selected triangle.
- Connect these three markings to each other to split the large triangle into four new triangles.
- Colour the newly formed triangle in the middle black.

The following pictures show the first iteration when there is only one large white triangle.



Question

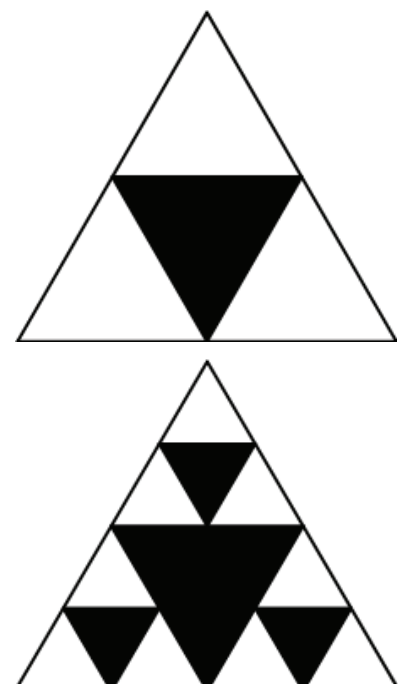
Click on the smaller triangles in the grid below to create the pattern that will be formed after the steps above have iterated three times. To save you some time, the first iteration has been done for you.



Answer

For the first iteration of the set of steps, the single centre triangle gets coloured black:

For the second iteration of the set of steps, every one of the three white triangles gets divided into four sub-triangles. The centre triangle of each of these four triangles gets coloured black. This does not change the original centre triangle, but three other newly coloured black triangles appear:



Continued on next page

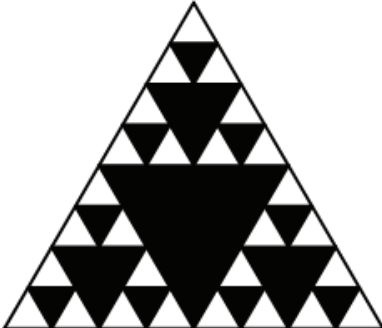




Sierpinski Triangle - continued

Answer - continued

For the third and last time that the set of steps are iterated, the same thing happens to every one of the 9 white sub-triangles. This creates $9 \times 4 = 36$ new sub-sub-triangles of which 9 are coloured black:



Therefore the image above is the correct solution.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Pattern Recognition, Abstraction, Modelling and Simulation, Algorithms

Concepts: Abstraction, Data Interpretation, Specification, Algorithms

The Sierpinski triangle is a fractal first described by the Polish mathematician Wactaw Franciszek Sierpinski (1882–1969) in 1915. Similar geometrical objects like the Koch snowflake or the Mandelbrot Set have been researched in the 20th century, especially since with computers an automated generation of different iterations was made possible. Because the process of dividing and colouring the triangles is supposed to be repeated infinitely, it offers some interesting geometrical properties, most notably a striking self-similarity.

The construction of a Sierpinski triangle is recursive (from the Latin term *recurrere*: to run back, to return). That means that a certain set of operations (the “steps” from above) are repeated infinitely often or until a certain termination condition (for instance how many times the iteration process is repeated) is met. This allows for very compact definitions for very complex objects.

Recursion is also a concept that is widely used in Computer Science, in theoretical concepts as well as in practical applications. The concept of a Sierpinski triangle is directly connected to classical Computer Science concepts like cellular automata (made popular by John H. Conway, 1937 – 2020) or the Towers of Hanoi.

If the construction of a Sierpinski triangle is repeated infinitely, one will end up with a completely black triangle in the end. That is because for each step only 75% of the originally white area stays white. This can be formulated as a sequence of how much of the originally white area is still white.

The sequence would be $a_n = 100\%, 75\%, 75\% \times 75\% = 56.25\%, \dots$, which is a geometric sequence with $a_1 = 1$ and $q = 0.75$. Since geometric sequences with $-1 < q < 1$ have a limit 0, in this case too we have $\lim_n \rightarrow \infty (1 \times 0.75^n) = 0$.



Passwords

A group of beavers create a set of passwords for securing their lodge. The passwords consist of only these two symbols: ★ ☾

A password checker is used to make sure that a given password is acceptable. The beavers use circles and arrows to describe how the checker works:

- The checker reads in the symbols from the provided password, symbol by symbol, from left to right.
- The checker always starts at the circle “S”.
- At each circle, the checker reads one symbol.
- If the symbol entered matches a symbol on an arrow that points from the current circle to another circle, the checker follows that arrow. Otherwise the checker stops and does not accept the password.
- Once all of the entered symbols are checked, the checker stops.
- If the checker stops on circle E, the provided password is accepted. Otherwise the password is rejected.

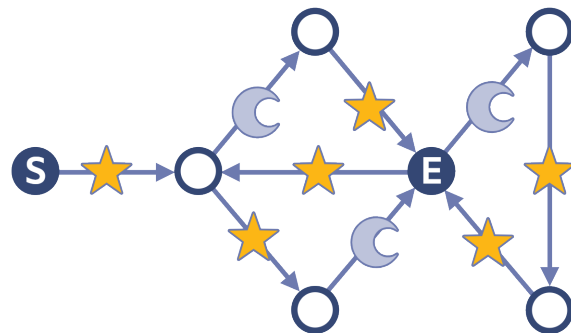
Example

Checker:



Entered password that will be accepted: ★ ☾ ★

The beavers create a new password checker:



Question

Select all of the following passwords that the new password checker will accept.

★ ★ ★ ☾ ★ ★ ★ ★ ★ ☾

★ ★ ☾ ☾ ★ ★ ★ ★ ☾ ★

☾ ★ ★ ☾ ★ ★

★ ★ ☾ ★ ☾ ★ ★ ☾ ☾ ☾

★ ★ ☾ ☾ ★ ★ ☾ ★ ☾





Passwords - continued

Answer





Answers A and B are correct.

For both passwords, the password checker stops at circle E after processing the last symbol.



Answer C is incorrect: The first symbol is  but all accepted passwords must begin with .

Answer D is incorrect: It consists of 13 symbols, but the length of all accepted passwords is divisible by 3.

Answer E is incorrect: Symbol  appears 7 times, and symbol  appears 5 times. But accepted passwords consist of twice as many  as .

It's Computational Thinking

Computational Thinking Skills: Pattern Recognition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Collection, Data Representation, Data Interpretation, Specification, Algorithms, Digital Systems

The password checkers shown in this Bebras task are modelled as deterministic finite-state machines (FSM); this is a mathematical model of computation. It is an abstract machine that can be in exactly one of a finite number of states at any time. The FSM can change from one state to another in response to some inputs; the change from one state to another is called a transition. An FSM is defined by a list of its states, its initial state, and the inputs that trigger each transition.

The behaviour of (finite) state machines can be observed in many devices in modern society that perform a predetermined sequence of actions depending on a sequence of events presented to them. Examples include:

- Vending machines, which dispense products when the proper combination of coins is deposited.
- Elevators, whose sequence of stops is determined by the floors requested by riders.
- Traffic lights, which change sequence when cars are waiting.
- Combination locks, which require the input of a sequence of numbers in the proper order.



Tree Sudoku

A beaver's field is divided into 16 plots arranged in a 4 x 4 grid where they can place one tree in each plot.

They plant 16 trees of heights 1, 2, 3, and 4 in each field by following the rules:

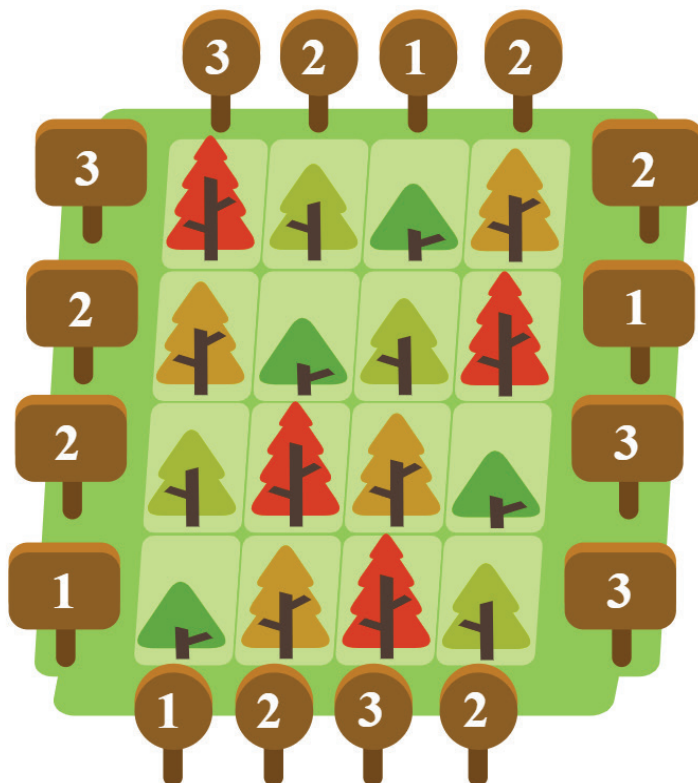
- Each row (a horizontal line) contains exactly one tree of each height
- Each column (a vertical line) contains exactly one tree of each height

If beavers observe the trees in a line, they cannot see trees that are hidden behind a taller tree. At the end of each row and column of the 4 x 4 field the beavers placed a sign and wrote on it the number of trees visible from that position.

Kubko has written down the numbers on the signs correctly but he placed some trees in wrong plots.

Question

Can you find the mistakes Kubko made and correct the heights of the trees?



Answer

When looking at Kubko's field, the placement of the trees follows the rules (each row and each column contains four trees of all four heights) but the numbers on the signs do not correspond to the numbers of trees visible from each position.

To solve this, first identify any rows or columns that are correct: where the numbers on the signs match the visible trees. Rows 2 and 3; and columns 2 and 4 should not be changed as they are correct. Any rows or columns that have the incorrect number of trees visible from the signs will need to have some tree sizes changed. Rows 1 and 4 and columns 1 and 3 have signs showing the incorrect number of trees seen, and thus must have incorrectly placed trees.

Using this logic, we can identify the plots which are at the intersection of these rows and columns that need to be changed. Changing only the height of these trees with mistakes means that the correct rows and columns remain unchanged.

Continued on next page



Tree Sudoku - continued

Answer - continued

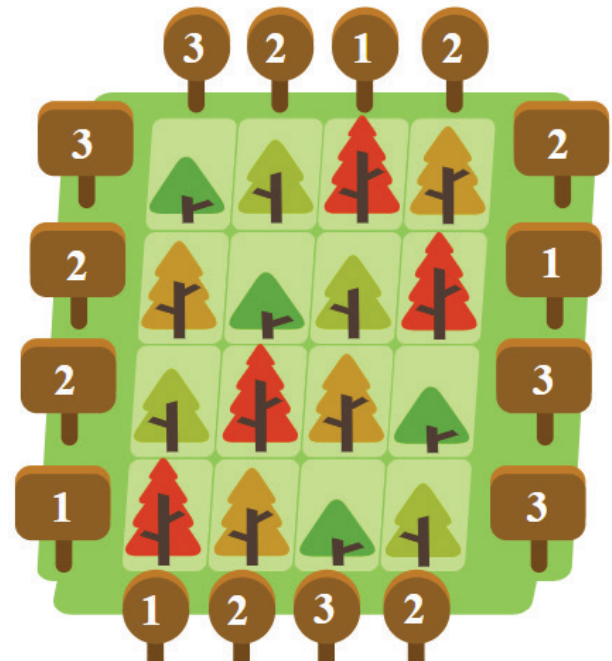
In the first column, swap the trees at the intersections. This will ensure the signs read correctly. Do the same in the third column.

By changing the trees in these four positions until all rows and columns obey the rules (each row and each column contains four trees of all four heights) and are the correct height for the signs, you can solve this problem.

It's Computational Thinking

Computational Thinking Skills: Abstraction, Algorithms, Evaluation

Concepts: Abstraction, Data Collection, Data Representation, Data Interpretation, Specification, Algorithms



This task approaches two fundamental skills for computer scientists. One is to find the solution for a problem that satisfies given constraints. The second one is the ability to reconstruct an object from partial information using knowledge about the properties of the object. This can be used for a compressed representation of objects. Moreover, one has to be able to follow rules and search for mistakes in data representation.

Error correction is a common technique used in Computer Science to ensure the reliability of data moving through channels which may lose or corrupt data. Error Correction techniques involve reconstructing incorrect data to its original and correct form.



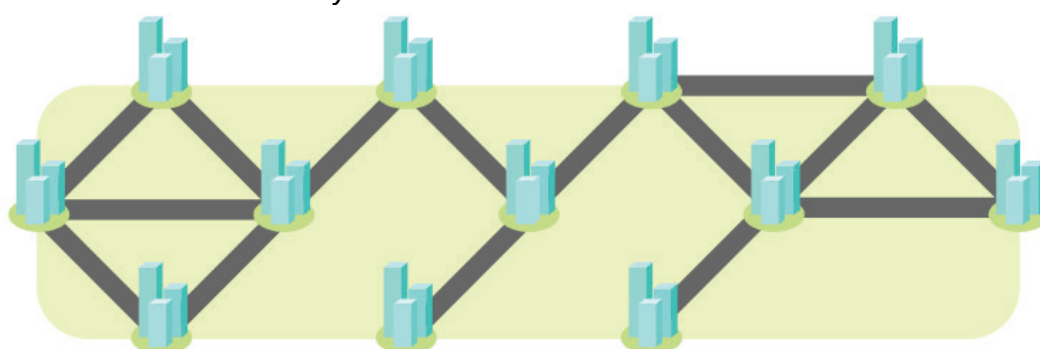
Epidemic Crisis

In Beaverland, there are 12 towns connected by highways as shown on the map below. Towns that are connected directly or indirectly by one or more roads form an 'economic community'. Currently, all 12 towns belong to the same community.

Unfortunately, due to an epidemic outbreak, the mayors have decided, in order to reduce travel between towns, to close two highways (using roadblocks). Their goal is to split the country into three separate economic communities. As they want to minimize economic disruption, once the roadblocks are in place, the smallest of the three resulting economic communities should contain as many towns as possible.

Question

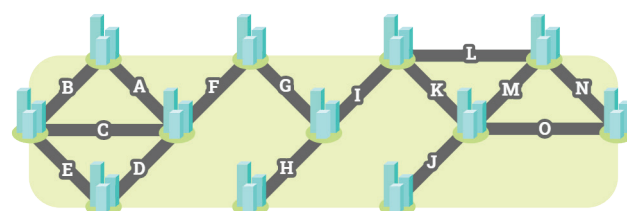
Click on the two roads they should close.



Answer

Using the lettering in the image below, the correct highways to close is "FI".

Firstly, there are some highways that will not split the economic community when blocked, so we first need to test which roads will split the economic community when blocked. This means that when a road is blocked, the endpoints of the road cannot be connected to each other by any other direct or indirect routes. When we test this, only five roads, F, G, H and I will meet this requirement.



Next, test all possible combinations of any two roads out of the five candidates, and check the size of the smallest block.

For example, if we block road F and G, the three communities will be 4, 1, and 7 in size. Thus, the smallest block size is $\min(4, 1, 7) = 1$.

Similarly, blocking any of these combinations of roads will also lead to a smallest community size of 1: (F, H), (F, J), (G, H), (G, J), (H, I), (H, J), or (I, J).

In another example, if we block roads G and I, the three communities will have sizes of 5, 2, and 5. Thus, the smallest community size is $\min(5, 2, 5) = 2$.

Lastly, if we block road F and I, the three communities will have sizes of 4, 3, and 5. Thus, the smallest block size is $\min(4, 3, 5) = 3$.

In this task, we want the smallest community to be as big as possible; therefore, blocking roads F and I is the best solution.

Continued on next page



This question comes from
Taiwan

Years 3+4

Years 5+6

Years 7+8

Years 9+10 Medium

Years 11+12



Epidemic Crisis - continued

It's Computational Thinking

Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Collection, Data Interpretation, Specification, Algorithms, Digital Systems, Impacts

This is a connected component problem. Connected component refers to situation in an undirected graph which any two vertices are connected to each other by paths within the graph. Graph traversal is used for finding connected components, starting from a source vertex and systematically following other vertices connected to the source vertex until no other vertex can be reached. In this problem, we also want to find bridges. A bridge is the edge which will cause the number of disconnected components to increase if the edge is removed.

Connected component analysis is used in multiple computer vision applications to identify the region of interest, such as character segmentation in OCR, foreground-background segmentation in visual tracking, and medical image processing.



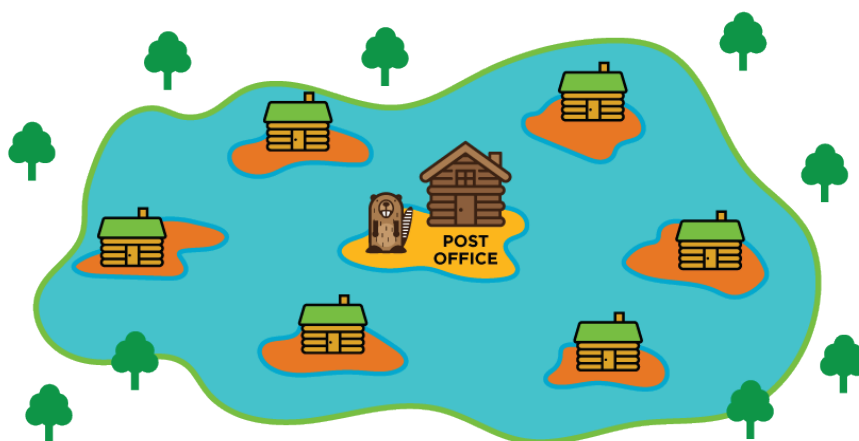
Aggelos the Mailman

A town on a lake has 6 houses where beavers live together or separately.

Aggelos is the new postal worker who doesn't know anything about the beavers who live in the town. Aggelos does not live in any of the 6 houses on the lake.

At the beginning, Aggelos' notebook is empty. He comes up with the following strategy to deliver the mail:

1. Every time a new letter is sent, Aggelos writes down the name and the address of the sender.
2. If the name of the recipient is in his notebook, he delivers the letter.
3. If the recipient's name is not in his notebook, he makes copies of the letter and delivers them to every house in the lake except to the house of the sender.
4. In any case, the correct recipient always replies to the letter the same day. Aggelos then writes down their name and address and delivers the reply letter.



Question

On the first day Elia sends a letter to George and Mike sends a letter to Elia.

On the second day Socrates sends a letter to Nasia.

How many letters has Aggelos delivered in his first two days of work?

Answer

Correct answer is 14 (delivered letters).

First, Elia sends a letter to George. Aggelos does not know where George lives so he delivers a copy of the letter to all of the houses except for Elia's (5 letters). George replies and Aggelos delivers the reply to Elia (+1 letter). Then Mike sends a letter to Elia. Aggelos knows where Elia lives so he delivers the letter directly to her (+1 letter). Elia replies the same day and Aggelos delivers it directly to Mike (+1 letter). In total he delivers 8 letters during his first day.

On the second day, Socrates sends a letter to Nasia. Aggelos does not know where Nasia lives so he delivers a copy of the letter to all of the houses except for Socrates' (+5 letters). Nasia replies (+1 letter). In total he delivers 6 letters during his second day.

Altogether, Aggelos delivered 14 letters in his first two days of work.

It's Computational Thinking


Computational Thinking Skills: Decomposition, Abstraction, Algorithms




Concepts: Abstraction, Specification, Algorithms

The procedure mentioned above is the method that a network switch uses to fill its MAC Address Table.



Heavy Parts

Sitting at their own tables, 24 parts engineers each build a part for a machine. The 24 parts are put together by an assembly engineer at table 'X'  to make the machine. The 24 parts engineers all start building their parts at the same time.

Seven heavy parts need to be transported to table  by a trolley-robot that takes 1 minute to go from one desk to the next. The robot needs to be charged after 16 minutes. The trolley-robot can start at any table, but must end at table . The time it takes to build the seven heavy parts is indicated on their tables (in minutes) e.g. .

- The path must allow the trolley-robot to collect the maximum number of heavy parts possible in one journey.
- Your path must consist of no more than 16 sections.
- No table can be visited more than once (so your path cannot cross over itself).
- Tables with heavy parts cannot be visited until the part is built.
- Your trolley can start at any table (that is not building a heavy part) but must end at table 'X'.

Question

What is the maximum number of heavy parts that the trolley-robot can collect in one 16-minute journey?

Answer

The correct answer is that you can receive, at most, 5 parts.

Not 7

There are 7 parts in total. But we can see easily that to collect all 7 isn't possible. Looking at the 2 tables with 12 and 15 (the tables where it takes the longest to produce a part), if a part is collected after 12 minutes from the (12) table, it takes 5 more minutes to get to the table with 15 on it. (the part at the table with 15 cannot be collected any earlier than a minute before the robot needs to be charged.) This cannot be done in the 16 allowed minutes.

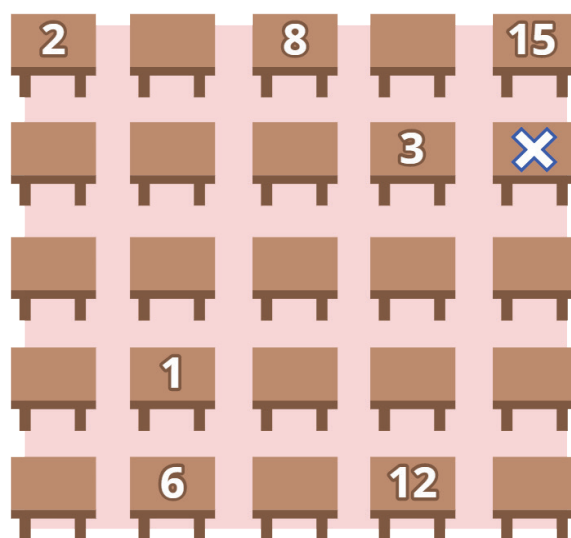
Not 6

So we know we cannot have both the 12 and the 15. But to have 6 parts in total, you must include either the 12 or the 15.

If we have either the 12 or the 15 we can get the parts back to the X table in the 16 minutes as the 12 table is 4 away and the 15 minutes is 1 away. So now the problem becomes proving that we can get to the 12 table, collecting all other parts in exactly 12 minutes (or the 15 table in exactly 15 minutes) Let's look at the 12 table. The next highest number of minutes to wait is 8. So if we could collect the parts from 1,2,3 and 6 in exactly 8 minutes, unfortunately it takes another 5 minutes to get to the 12 table, which is one more than we can allow.

So we look at the 15 table and see if we can collect the parts from 1,2,3,6 and 8 in 13 minutes. (Another 2 to get to the 15 table from 8 or 3) This is a bit harder to prove by elimination. There are 120 permutations and it's not possible to explore them all in this explanation.

Hopefully it can be seen that waiting near the 1 desk and then collecting that part will waste time as the nearby 6 cannot yet be collected and so the robot will need to return to that end of the room at some stage.





Heavy Parts - continued

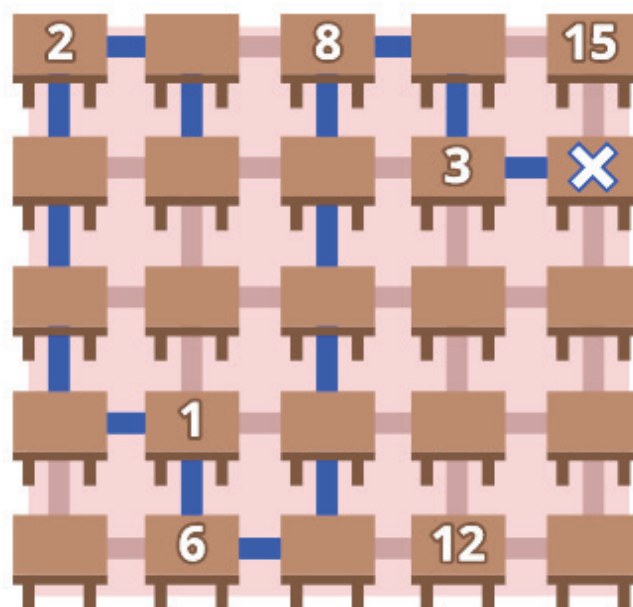
Answer - continued

Similarly the 3 requires waiting for 3 minutes, then wasting time waiting for the nearby 8, or returning to that part of the room. So we don't want the robot to have to "return" to the area where the 1 and 6 are or "return" to the area where the 3 and 8 are. So if the robot starts near the 2 and then collect that after 2 minutes. If we go to the 3/8 area, then the 1/6 area then back to the 15, the robot has traversed the room several times and exceeded the time limit.

So the robot would be best served collecting the 2 and going to the 1/6 area and then the 3/8 area. The robot takes 2 minutes at the 2 desk (2 minutes). It takes 4 minutes to travel to the 1 desk (6 minutes). Another 1 minute is taken travelling to the 6 desk (7 minutes). 5 minutes is needed to travel to either the 3 or 8 desk (12 minutes) and then 2 minutes to the other of those two desks (14 minutes) and now from either the 3 or the 8 desk takes 2 minutes to travel to the 15 desk and now the 16 minutes have been used up and the robot has not returned to the X desk.

But 5

From the explanation above we can see that we can get to the 3 table in 14 minutes collecting 2,1,6,8,3 in 14 minutes and it's only a further 1 minute to the X desk. For 5 solutions several possibilities exist, one of which is shown in the diagram.



It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Collection, Data Interpretation, Specification, Algorithms, Implementation, Digital Systems, Interactions, Impacts

This is an optimization problem. In this case the number of tables visited should be maximized. The problem also has some additional constraints: the amount of time it takes before a solution is found. The way that the number of tables is counted is by creating a path through the graph (as shown in the answer explanation). In this case the tables are represented as nodes and the connections between tables as edges. A path is a sequence of edges in the graph. The number of minutes it has taken to go from the first node to the last node is written down for each edge of the path. A path is a valid solution of your problem, if the number of an edge is at least one more than the number of the node it is leading from.

From all valid paths the maximum number of numbered tables is searched.

This type of problem can be solved by different methods. The answer explanation uses contradictions to show that valid paths that visit 6 or 7 numbered tables do not exist. And it gives a single example that proves that a solution for 5 numbered tables exists. This is a formally correct way to show that a solution is correct.

When first being confronted with a problem, however, most people take a constructive approach. They try to construct a valid path, maybe run into a problem, move back a few steps, try a different approach, until they have an idea what could be true or not. This method is called branch & bound and was first conceived by Ailsa Land and Alison Doig in 1960, because one for each step four different next steps are possible and one decides to try one and if it's not leading to a success, it is no longer considered. This method is commonly used for inherently complex problems in Computer Science.



Math Machine

The beavers created a MathMachine. It takes a number as input and returns another number as output. Inside the MathMachine, there are many components. All components work in the same way. Each component takes three numbers as input and processes them as follows:

If the first number is 1, return the third number as the output of the MathMachine.

If not:

- Decrease the first number by 1. The result is the new first number.
- Increase the second number by 2. The result is the new second number.
- Add the new second number and the third number. The result is the new third number.
- Pass the new numbers to the next component, in the same order.

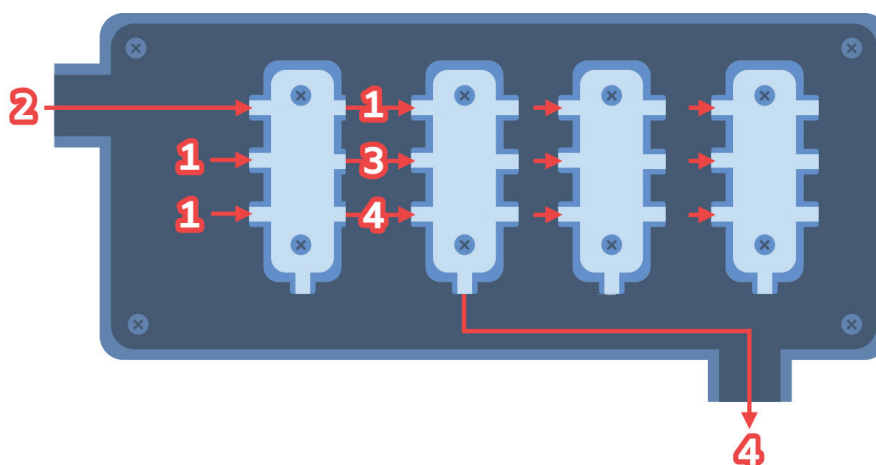
The first component is special:

- When the MathMachine receives an input, it passes this number as the first input to the first component.
- The other two inputs for this component are 1 and 1.

As soon as the MathMachine receives an output from any of its components, it returns this number as a result.

Example

The image shows how the MathMachine processes the input 2. This example uses only two of the MathMachine's many components.



Question

The MathMachine processes the input 4. Which number does the MathMachine return as output after passing through all four components?

Answer

The answer is 16.

The initial component receives the inputs (4, 1, 1) - processing the first component, we subtract 1 from the first number, add 2 to the second number, then add the second number to the third to get the new third number. This gives us the inputs for the second component, which are (3, 3, 4). Undertaking the same process for the second component, we get input for the third component of (2, 5, 9). We put these numbers through the same process for the third component, we get (1, 7, 16).

We further know that when the component receives a 1 as the first number, it outputs the third number as the final output for the whole machine. This means that when the fourth component processes the input (1, 7, 16) it outputs the third number, which is 16.

Continued on next page



This question comes from
Germany

Years 3+4

Years 5+6

Years 7+8

Years 9+10 Hard

Years 11+12



Math Machine - continued

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms

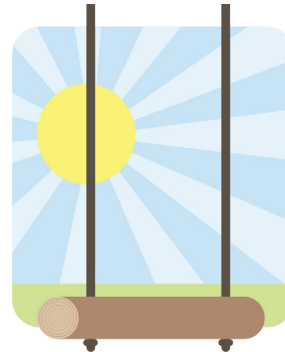
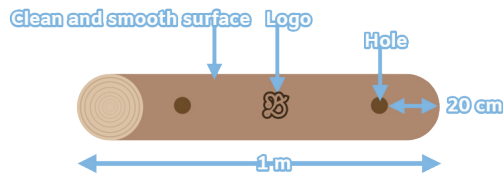
Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Implementation, Digital Systems

A common method of simplification is to divide a problem into subproblems of the same type. As a problem solving technique, this is called divide and conquer and is key to the design of many important algorithms. Recursion in Computer Science is a method of solving a problem where the solution depends on solutions to smaller instances of the same problem. Recursion solves problems by using functions that call themselves from within their own code. The approach can be applied to many types of problems, and recursion is one of the core ideas of Computer Science.



Wood Processing

A delivery of 100 wooden logs has arrived. All the logs are longer than 1m.
A team of four robots process the logs. Their task is to produce a full set of swing seats, as shown in the images below.



All four robots are able to select logs that still need to be processed and then perform their individual function on the selected log. The robots can work at the same time as each other if there are logs they still need to process:

Cutter: Cuts a log on one side and makes it 1m long.

Driller: Drills two holes through the log exactly 20cm from the left and 20cm from the right.

Printer: Prints the company's logo in the middle of a smooth and clean log surface.

Remover: Removes the bark from logs of any length and makes the surface smooth and clean.

- A robot processes each log only once.
- The robots can work at the same time as each other but not on the same log.
- A robot starts working when it gets a start signal from the Control Program.
- From the moment a robot is started, it must never be idle, waiting for logs from other robots.
- A robot stops working once it has processed all logs, or if there are no more logs to process.

The commands in the Control Program are limited to those given below and are started sequentially.

Question

Rearrange the commands in the Control Program so that the robots process all the logs without any of them ever being idle.

Start the Remover

Start the Printer

Start the Cutter

Wait until all working robots have finished their work

Start the Driller

Answer

There are multiple correct answers to this question. Here is a possible solution:

The important part is the placement of the wait comment. The two commands before the wait command can be given in any order and the commands after the wait command can also be given in any order without changing the effect of the program.

Start the Cutter

Start the Remover

Wait until all working robots have finished their work

Start the Driller

Start the Printer



Wood Processing - continued

Answer - continued

The Cutter and the Remover must complete their work first, and the Driller and the Printer start working after the other two have finished. Both the Driller and the Printer require the Cutter to have completed its work, so that the logs are the correct length (1m long). If the logs had not already been cut to the incorrect length, the placement of holes and printing of the logo would be in the incorrect positions once the logs had been cut. The Printer further needs logs with a smooth and clean surface, which tells us that the Remover needs to operate before the wait command as well.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Collection, Data Interpretation, Specification, Algorithms, Implementation, Digital Systems

This task illustrates two basic techniques of parallel computing:

1. Mapping
2. Process synchronising

1. Mapping is an alternative concept to iteration. You define a function and map it on a collection. That means, the function is “applied on” each element of the collection but you do not care in which order this takes place. Since the function can be executed concurrently by several processors, mapping can be faster than iteration. An example of this using Python is:

```
>>> fruits = ['apple', 'orange', 'cherry']
>>> s = map(str.upper, fruits)
>>> print(list(s))
['APPLE', 'ORANGE', 'CHERRY']
```

Here the string method `upper()` is mapped on a list of strings.

2. The robots in the task could work concurrently on the collection of logs. The control program made sure that the Driller and the Printer did not start before Cutter and Remover had finished their work. This is called process synchronization. In a computer several processes can work concurrently on the same data.



Towers of Blocks

Sam the beaver is playing with his toy blocks. Each block is the same height, but has a different width. He built nine beautiful towers, each one made with blocks of the same width.

There are two ways to change the height of a tower: adding blocks to the top or removing blocks from the top. Either way, the energy cost to change the height of a tower is proportional to the width and the number of blocks being changed.

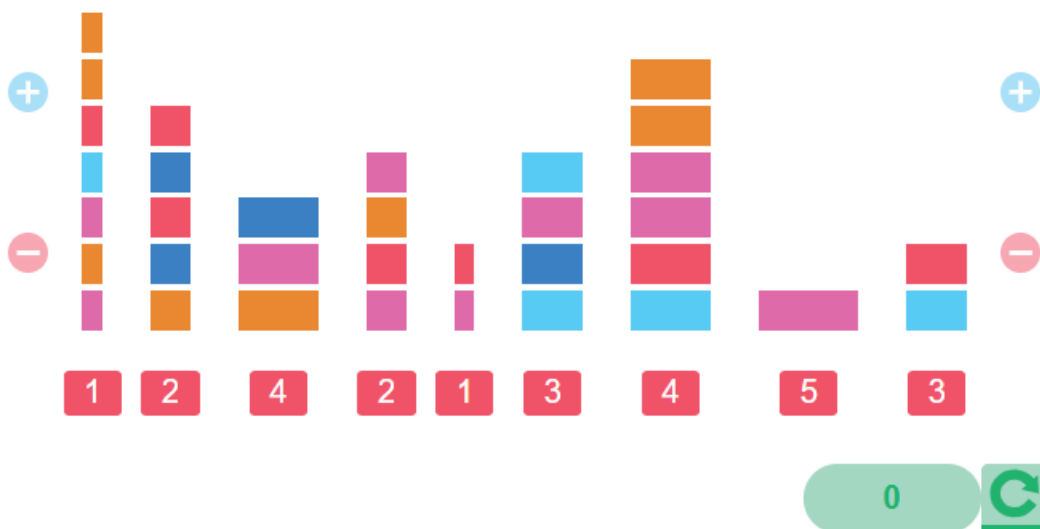
Example

Removing 2 blocks from a tower with a width of 1 costs $2 \times 1 = 2$ units of energy; and adding 4 blocks to a tower with a width of 3 costs $4 \times 3 = 12$ units of energy.

Sam wants all towers to be the same height, and he wants to spend as little energy in total as possible.



Question

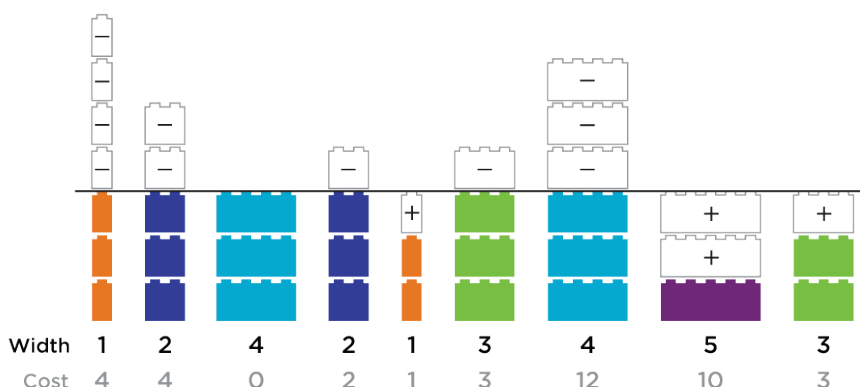
In total, what is the minimum amount of energy that it will cost Sam to make all towers the same height?



Answer

The correct answer is 39.

The following figure shows the correct solution, using  to show blocks that have been added and  to show blocks that have been removed, making all the towers 3 blocks tall:





Towers of Blocks - continued

Answer - continued

The median height of the all nine towers is 4 (1 2 2 3 4 4 5 6 7). One might think the median height is the answer, but this is not the case here.

This is due to the fact that the blocks do not have uniform width. However, the function determining the cost is unimodal (has a single minimum), so instead of trying all possible heights, we can use a smarter strategy for finding the minimum such as ternary search. By applying ternary search, we can eliminate the two highest heights and the two lowest heights out of seven heights, leaving height 3, 4, and 5. Then we only need to calculate the energy cost of forming these three heights.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Evaluation

Concepts: Abstraction, Data Collection, Data Interpretation, Specification, Implementation

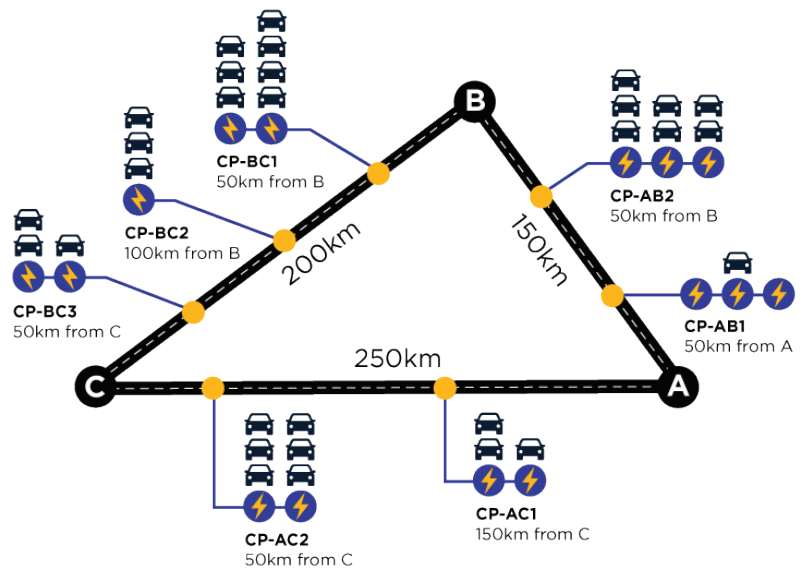
Each pile of blocks is what we call a stack in Computer Science, with two possible operations: push (putting a new block on the top of the stack) and pop (removing the topmost block). In this problem, we need to take into account the current height and width of each tower already built. Stacks are very useful in many algorithms more complex than basic block building. For example, they can be used to evaluate arithmetic expressions.

Searching for the best solution (such as the minimum cost) among all possibilities is a very common task in Computer Science. This is known as an optimization problem. In this case, because the cost has a unimodal distribution, we could use ternary search, which determines that the minimum or the maximum can be in neither the first third nor the last third of the domain, and then repeats on the remaining third. A ternary search is an example of a divide and conquer algorithm.

Electric Car Queues

You have an electric car that travels at 100 km per hour and can travel 200 km when fully charged. The car can be recharged at any time before the charge runs out. Each recharge restores the car to full charge and takes 1 hour, regardless of how much charge is left when it starts recharging.

The picture below shows the roads connecting A, B and C. You start at point A and want to visit points B and C in as little time as possible. You can take any route — via B to C, or via C to B. When you pass through point B or point C, you must stop for 30 minutes, but your car does not lose charge.



Along the route there are charging stations, each with a certain number of charging points. At each charging point, there may be cars currently in queue, and you have to wait for them to finish charging before you can use this charging point. For instance, at the charging station CP-AB1 there are three charging points, two of which are empty and one which has a single car in queue. Each car waiting in queue takes 1 hour to charge. The first car in each queue has just started charging when you set out from point A. You start with your car fully charged.

Question

What is the least amount of time it will take to visit points B and C, starting at point A?

5 Hours

6 Hours

6 Hours 30 Minutes

7 Hours

Answer

The correct answer is 6 hours.

You can achieve this by choosing the route A to B to C as follows. Start at A and drive 50 km to CP-AB1 (30 mins). Recharge at CP-AB1 (1 hour). Drive 100 km to B (1 hour). Wait at B (30 mins) with 100 km worth of charge remaining. From B, drive 100 km to CP-BC2 (1 hour). By now you have travelled 4 hours and all cars in the queue at CP-BC2 have finished charging. Recharge at CP-BC2 (1 hour). Drive 100 km from CP-BC2 to C (1 hour). Total time spent is 6 hours.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Digital Systems, Interactions, Impacts

Finding the shortest route/path between a set of points is a common problem in Informatics. Finding such paths make it possible to efficiently transport people and physical goods from one place to another, as well as routing electronic packets of information to their destination on the Internet as fast as possible.

If you go from A to B via C, you travel 450 km (4 hours, 30 minutes) and spend 30 minutes at point C. In addition, you must recharge at least twice (2 hours), so the minimum time for this route is 7 hours, assuming you always recharge without waiting.



Bebras Challenge 2021 Round 1

Years 11+12



Don't Crash

A robotic vacuum cleaner robot moves in a room of 6×7 square-shaped tiles with walls surrounding it. The robot is always in the middle of a square and is always facing one of the four walls. The robot can be programmed to move using these commands:



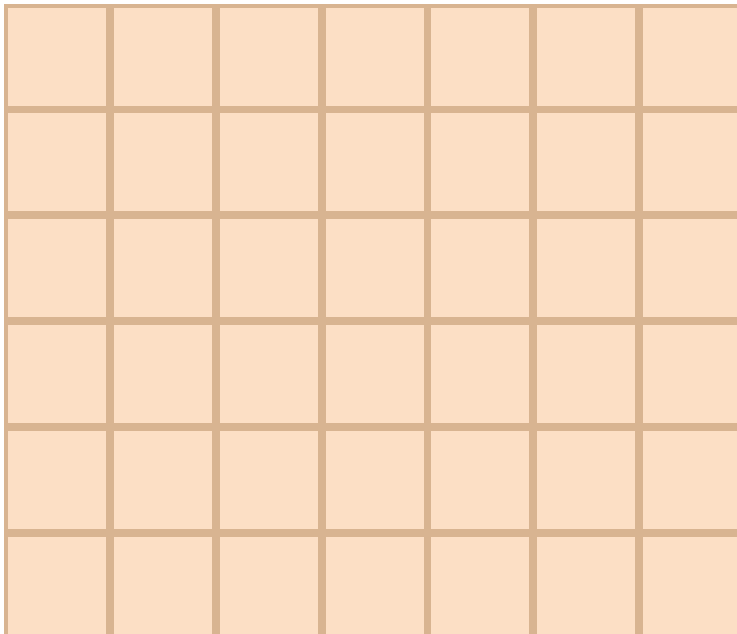
- **FORWARD:** move to the next tile it is facing.
- **LEFT:** turn 90 degrees counterclockwise while staying on the same tile.
- **RIGHT:** turn 90 degrees clockwise while staying on the same tile.

The robot is about to execute the following program:

FORWARD LEFT FORWARD RIGHT FORWARD

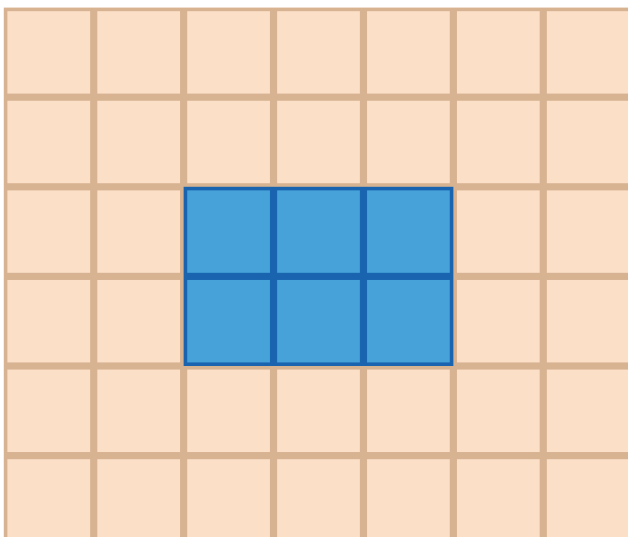
Question

In the room map below, select all the tiles where the robot can start and execute this program without crashing into a wall, no matter which wall it is initially facing.



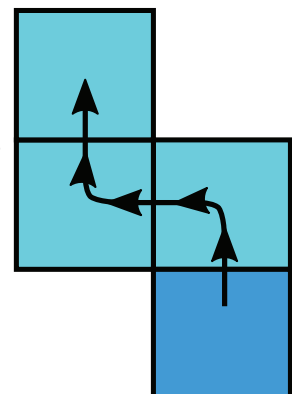
Answer

The correct answer is shown in the picture below.



During execution of the program, the robot moves two tiles forward and one tile left:

For this movement to be possible regardless of the direction the robot is pointing, the robot cannot start on any of the squares directly touching the walls, since there is always one direction where it would immediately run into a wall. Likewise, it cannot start on any of the squares that are touching those squares, since there would be one direction where it would run into a wall



Continued on next page



Don't Crash - continued

Answer - continued

after moving forward once. The highlighted squares in the first image are far enough away from the walls that no matter what direction the robot is facing, it will not run into a wall making this movement.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms

Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Implementation, Digital Systems

When writing a program, programmers need to think about situations that will be encountered as a program executes. They need to take into account things that might lead to crashes, or to an impossible behaviour for the machine executing the program. They must check and define the conditions that the program will work under. They must be able to test the program.



Stickers

Betty Beaver is playing with four types of stickers, which contain the words ABBA, GAGA, GIBB and IGGY. She creates a word by putting the stickers on an empty piece of paper.

When a sticker is placed at a given position, it covers the character in that position and the three characters after it. Betty has multiples of each type of sticker.



Example

One way to create the word GIABIGGYGA would be to use Betty's stickers as follows (asterisks mean empty positions):

1. GIBB at position 1: GIBB*****
2. ABBA at position 3: GIABBA****
3. GAGA at position 7: GIABBAGAGA
4. IGGY at position 5: GIABIGGYGA

Question

From the following four words, select all the words that can be created using Betty's stickers.

AGGIBBAGGAGABABGA

AGGIBBAGAGGABABGA

AGIBBGAGAGGYBAYBB

AGGIBBAGAGGYBAGGY

Answer

A), C) and D) can all be created using Betty's stickers.

To solve this task, it is easiest to start from the final word and move step-by-step until all the letters are gone. At each step we identify a sticker that was used (a substring of the current word that matches with one of the stickers) and remove it from the paper. A crucial observation is that if there are multiple options, it does not matter which sticker we remove as long as it is from Betty's collection,

Continued on next page



Stickers - continued

Answer - continued

we can reach an overall solution if one exists. When a sticker is removed, we no longer need to know what characters that part of the word contains. This means that as we progress, we can look at any sticker that overlaps with a previous sticker, no matter what the characters are in the overlapping part.

By applying this method, we can work backwards through each of the four different answer choices as shown in the table below. The sticker(s) being removed at each level are bold and underlined, and stickers are replaced with asterisks once they have been removed.

A	B	C	D
AG <u>GIBBAGGAGABABGA</u> AG****AG****BABGA AGAGAGAGA* <u>ABBA</u> BGA A*****BGA <u>ABBA</u> ***** <u>GIBBGA</u> *****GA ***** <u>GAGA</u> ***** Reached empty: OK!	AG <u>GIBBAGAGGABABGA</u> AG****AGAGGABABGA AGAGAGAGA <u>GGABABGA</u> A*****GGABABGA <u>ABBA</u> *****GGABABGA *****GGABABGA No more stickers, no solution exists!	AG <u>GIBBAGAGAGGYBAYBB</u> A*****GGYBAYBB <u>ABBA</u> ***** <u>IGGYBAYBB</u> *****BAYBB ***** <u>ABBAYBB</u> *****YBB ***** <u>IGGYBB</u> *****BB ***** <u>GIBB</u> ***** Reached empty: OK!	AG <u>GIBBAGAGGYBAGGY</u> AG****AGAGGYBAGGY AG <u>GIBBAGAGAGGYBAGGY</u> A*****GGYBAGGY <u>ABBA</u> ***** <u>IGGYBAGGY</u> *****BAGGY ***** <u>ABBAGGY</u> *****GGY ***** <u>IGGY</u> ***** Reached empty: OK!

It's Computational Thinking

Computational Thinking Skills: Decomposition, Pattern Recognition, Modelling and Simulation, Evaluation

Concepts: Data Interpretation, Specification

This is an example of a problem that can be analysed using “backward induction” and “backtracking”, which are common problem solving methods in Computer Science. Backward induction is a process of reasoning backwards in time: we start from the end of a problem or state (e.g. in this task a created word), and try to determine an action (in this task use of a sticker) that leads to a feasible preceding state. This is repeated until the initial state (in this task, a completely empty state) is reached.

In many problems the process may have several possibilities for selecting actions, and in such cases the backward induction process may need to be applied in a backtracking manner: if the currently selected sequence of actions fails to reach the initial situation, then we may change a previous action and again try to move towards the initial state.

Lemmings

The lemming King wants to send a message to his queen, who lives in another castle. He chooses four lemmings to deliver his message and gives each of them a flag which is either dark red or pale yellow, according to the message he wants to send.

But the King worries that something might go wrong during the journey, so he chooses three more helper lemmings and gives them flags following to these rules:

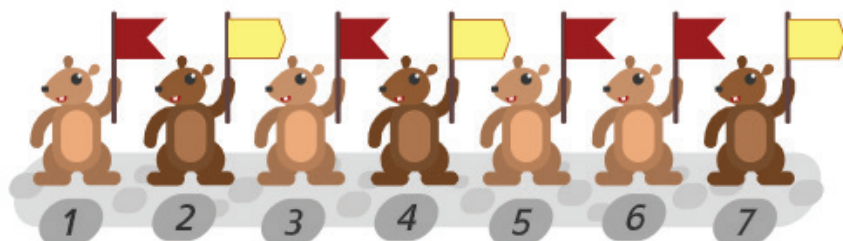
- **The 5th lemming is helping lemmings #1, #2, and #3:** If these lemmings are carrying an odd number of dark red flags, then lemming #5 will carry a dark red flag, otherwise he will carry a pale yellow flag.
- **The 6th lemming is helping lemmings #1, #2, and #4:** If these lemmings are carrying an odd number of dark red flags, then lemming #6 will carry a dark red flag, otherwise he will carry a pale yellow flag.
- **The 7th lemming is helping lemmings #2, #3, and #4:** if these lemmings are carrying an odd number of dark red flags, then lemming #7 will carry a dark red flag, otherwise he will carry a pale yellow flag.

On the journey, one of the lemmings lost his flag. To cover up his mistake, he quickly made a new one. Unfortunately, he doesn't remember which colour his original flag was, so he is not sure if his new flag is correct or not. When the lemmings arrived at the Queen's castle, she saw the messengers lined up as below.

Question

Exactly one lemming lost his flag, but we don't know if his new flag has the right colour.

Select the lemming that lost his flag, if it is possible to know.

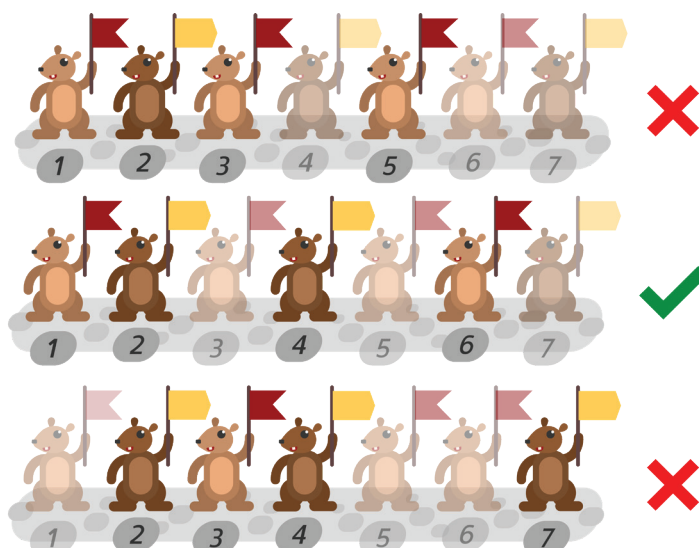


Answer

Lemming #3 has lost his flag, and the new flag has the wrong colour.

To work out the answer, we can group the lemmings into different groups, named after their helper lemming. So group #5 is lemming #5, along with lemming #1, #2 and #3

From the rules that the King used to hand out flags to the helper lemmings, we know the following: Before the journey begins, every group should have an even number of red flags, since if the number of red flags carried by the messenger lemmings is odd, the helper lemming will get a red flag, making it even. So if #1, #2 and #3 had an odd number of red flags, #5 would have to



Continued on next page



Lemmings - continued

Answer - continued

carry a red flag, making the total number of red flags in the group even. This is also true of group #6 and group #7. Now we can check to see if this is true for all the groups after the lemmings arrived at the Queen's castle.

Firstly, we can see that this statement is not true for all of the groups. This means that the lemming who lost his flag definitely made a flag of the wrong colour.

Secondly, if the lemming who lost his flag was one of the helper lemmings, only that group would be violating the rule. Since there are two groups that the rule is not true for, we know that it is one of the messenger lemmings who has lost his flag. Knowing that there is only one lemming who may be carrying the wrong coloured flag means that we are looking for a lemming that is in both of the groups that have violated the rule (#5 and #7), but is not in the group where the rule is true (#6). The only lemming that is in both group #5 and group #7 and is not in group #6 is lemming #3.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Algorithms, Evaluation

Concepts: Abstraction, Data Representation, Data Interpretation, Specification, Algorithms, Digital Systems

This task and the rules for helper lemmings described in it is an example of an error correction code in action. One of the first error correction codes is the Hamming code, which works similarly to the method used by the King in this task.

When we send messages through communication channels (wires or radio waves) in binary format, there is always a chance that some bits in the message might be flipped due to noise in the environment (changed from 0 to 1 or the other way around). We usually want to be able to detect if the message was altered and if possible determine what the original message was. Here is where error correction codes come in handy.

One simple way to check for errors is to send each bit in the message several times. For example, if we send every bit three times, then even if one of the three transmissions has a flipped bit, we will know what the original message was. We can send each bit more times to protect the message against even more flips. However, this approach requires us to send three or more times as much data with every message.

Fortunately, many smart algorithms allow sending as few extra bits as possible while still allowing the receiver to check and correct the message. One of these algorithms is used in this task. It is an example of a Hamming code. Hamming codes rely on the parity (evenness or oddness) of the sums of groups of bits in the message, including the extra bits. If the parities of all the groups are correct, then we can assume that the message arrived intact. If some of the parities are incorrect, then some bits (maybe the extra bits) were flipped during the transmission. If we know that at most one bit might be flipped, and we know which groups have the wrong parity, we can tell which bit was flipped and what the original value of that bit was.

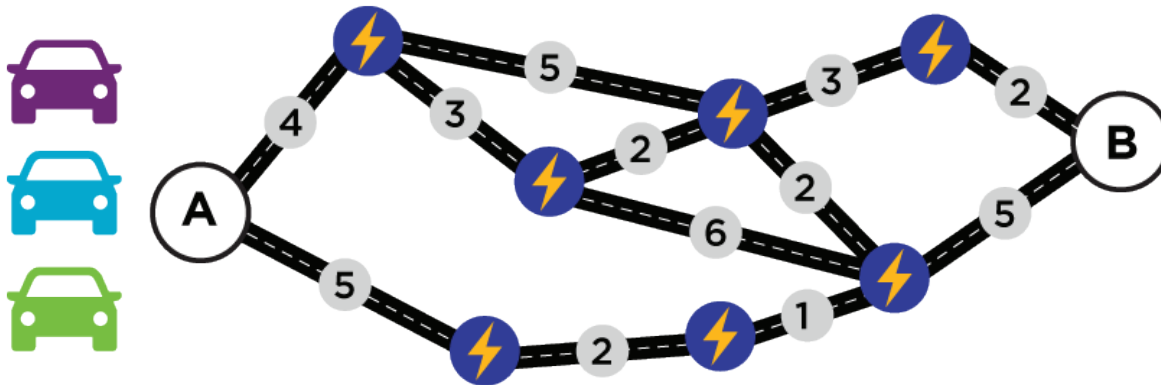
Every error correction code has its limitations. Some codes are designed only to detect that there was a change in the message without having a way to figure out which bit was changed. Some codes only work if no more than a specific number of bits were changed. For example, the code in this task only works if no more than one lemming wrongly replaced his flag. If two lemmings wrongly replaced their flags, the Queen would not be able to determine which flags had been changed. In the task, we assumed that only one lemming incorrectly replaced his flag. If we didn't know that, it might have been the case that the message arrived intact, even if the parities are wrong (for example, if both lemmings #5 and #7 changed their flags). It's fair to assume that it is most likely that only one or a small number of bits in a given message will be flipped, rather than a large number. When this assumption is correct, error correction codes like Hamming codes work well. However, real communication systems are usually more complicated than that, and more complicated codes have been developed to handle more difficult situations.

Similar algorithms are used to verify the accuracy of the information in other places, such as bar codes on various goods, and identification numbers on passports and IDs.

Electric Cars

The blue car can travel 4 km before it needs charging and needs 3 minutes to charge. The green car can travel 5 km before it needs charging and needs 4 minutes to charge. The purple car can travel 6 km before it needs charging and needs 5 minutes to charge.

A car does not need to stop at every charging station, but when a car stops to charge, it must wait for the full charging time, no matter how much charge the car has left. All cars travel at the same speed, travelling 1km every minute and start at node A fully charged.



The map above shows all the roads and charging stations between A and B. The numbers indicates the distance (in km) between each charging station.

Question

Based on the map, which car can travel from A to B in the fastest time?

Purple

Blue

Green

Answer

The correct answer is Green.

The green car can follow the route with distances {5, 2, 1, 5} which is a total of 13 km, taking 13 minutes. It only needs to stop at the first and third charging stations, taking an additional 8 minutes (2×4 minutes) for charging. This gives a total of $13 + 8 = 21$ minutes.

The fastest that the blue car can get to point B is 26 minutes by following the route with distances {4, 3, 2, 3, 2} which is a total of 14 km, taking 14 minutes. The car needs to stop at all the 4 charging stations, taking an additional 12 minutes (4×3 minutes) for charging. This gives a total of $14 + 12 = 26$ minutes.

The fastest that the purple car can get to point B is 23 minutes by following the route with distances {5, 2, 1, 5} which is a total of 13 km, taking 13 minutes. It only needs to stop at the first and third charging stations, taking an additional 10 minutes (2×5 minutes) for charging. This gives a total of $13 + 10 = 23$ minutes.

Continued on next page



Electric Cars - continued

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Digital Systems, Interactions, Impacts

Finding the shortest route/path between points is a common problem in Informatics. In modern day technology, we would want to be able to communicate with somebody else somewhere in the world as quickly as possible. Finding the shortest route between two points may ensure fastest communication.

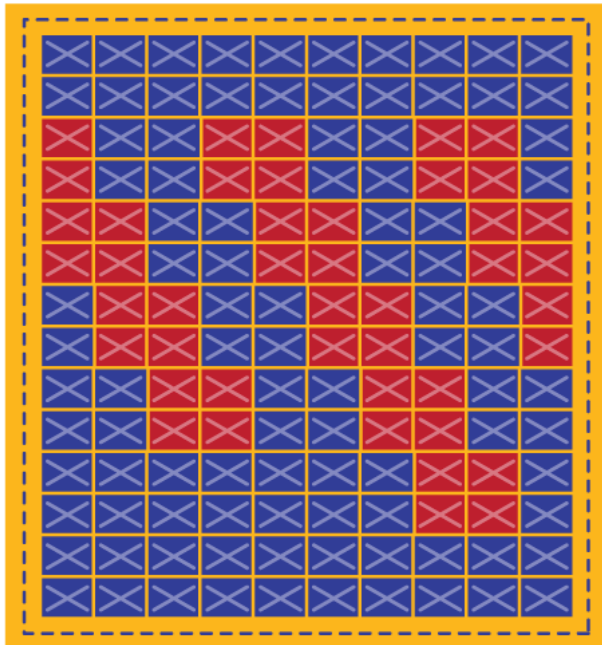
Various algorithms were developed over the years to aid in finding the shortest route in any network. One such algorithm was published by a Dutch computer scientist, Edsger Dijkstra, and is now known as Dijkstra's algorithm. This algorithm finds the shortest path from a source node to every other node.



Needlework

Lina loves Konavle embroidery. She wants to learn how to make some of the patterns. Her friend Tereza gave her a single instruction and provided some code to help her learn the patterns.

The instruction reads: “Look from the bottom up and from left to right”



Code

8.4.2

6.4.4

4.4.6

4.2.4.2.2

8.4.2

6.4.4

4.4.6

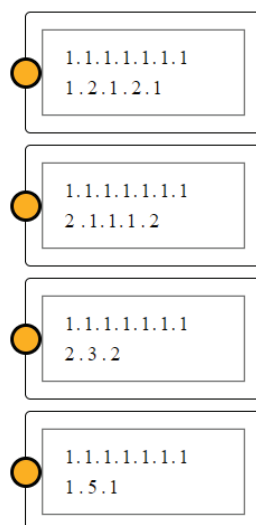
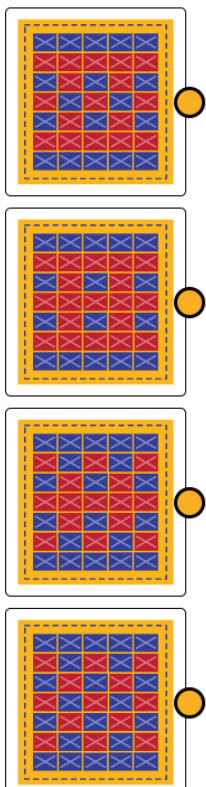
2.4.4.2.2

2.2.4.4.2

6.4.4

Question

Teresa gave Lina a number of blocks of code for new samples. As a computer scientist, she shortened the code blocks and wrote down only the part of the code that repeats. Drag a line between each code block and the matching pattern.





Needlework - continued

Answer

1.1.1.1.1.1.1	1.1.1.1.1.1.1	1.1.1.1.1.1.1	1.1.1.1.1.1.1
2.1.1.1.2	1.2.1.2.1	2.3.2	1.5.1

From the example, we can see that each row in the code represents one column on picture. It is important to follow Tereza's instructions and start reading the pattern from the bottom left corner, reading upwards, column by column. By examining the pattern, we can see that each new number changes the colour being stitched and the value tells us how many squares that new colour continues for.

When we apply this algorithm to the pattern, we find that the first row in all the patterns in the question are all the same. The differences appear in the second row of each pattern, which is where the code differs as well. Checking the patterns of the second column against the second line of each code block will give the answer.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Pattern Recognition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Representation, Data Interpretation, Specification, Algorithms

An algorithm is a set of instructions designed to perform a specific task or well-defined procedure that represents a solution to a problem.

It is a sequence of unambiguous instructions. 'Unambiguous' indicates that there is no room for subjective interpretation. Every time you ask a computer to carry out a specific algorithm, it will do it in exactly the same manner with the exact same result.

The act of representing of one object using a set of other objects is called encoding. Encoding is very common in Computer Science, and is used to represent a range of different data types.



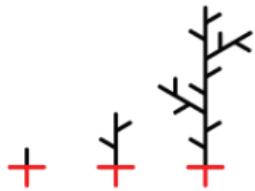
In this task we replace each number in code with the sequence of red or blue cells. Sequences of cells are separated by a decimal point. Cells can be one of two possible values: red or blue. When the instructions are followed correctly, the result is always identical.

Digital Trees

In Bebras Town you can grow digital trees!

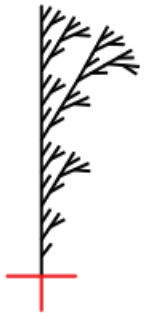
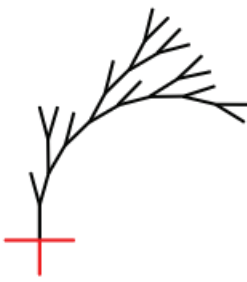
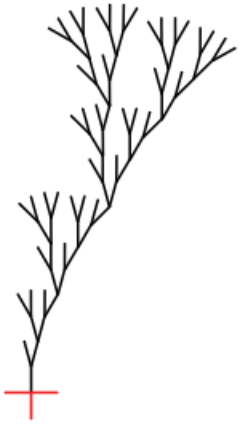

Each tree starts with a small branch and a rule. The rule tells you how the starting branch will be replaced. This step can be repeated as shown below.





Red lines indicate the starting point of the tree and are not changed by the branch rules.

Start	Rule	Result for 0,1,2 iterations	
			Remember, that every branch of the tree is replaced simultaneously by what is shown in the rule. The arrow symbol at the top of the rule is not inserted; it only shows where the rest of the tree is connected.

Question

Drag and drop each rule to the tree that is produced by that rule.






Answer

Rule number 1:

At each iteration, the rest of the tree is added to the right hand sub-branch and is pointing straight up. Result number 3 fits those criteria.

Rule number 2:

At each iteration, the rest of the tree is added at the centre branch and the right hand branch can never get longer than the centre branch. This only matches the result number 1.

Digital Trees - continued

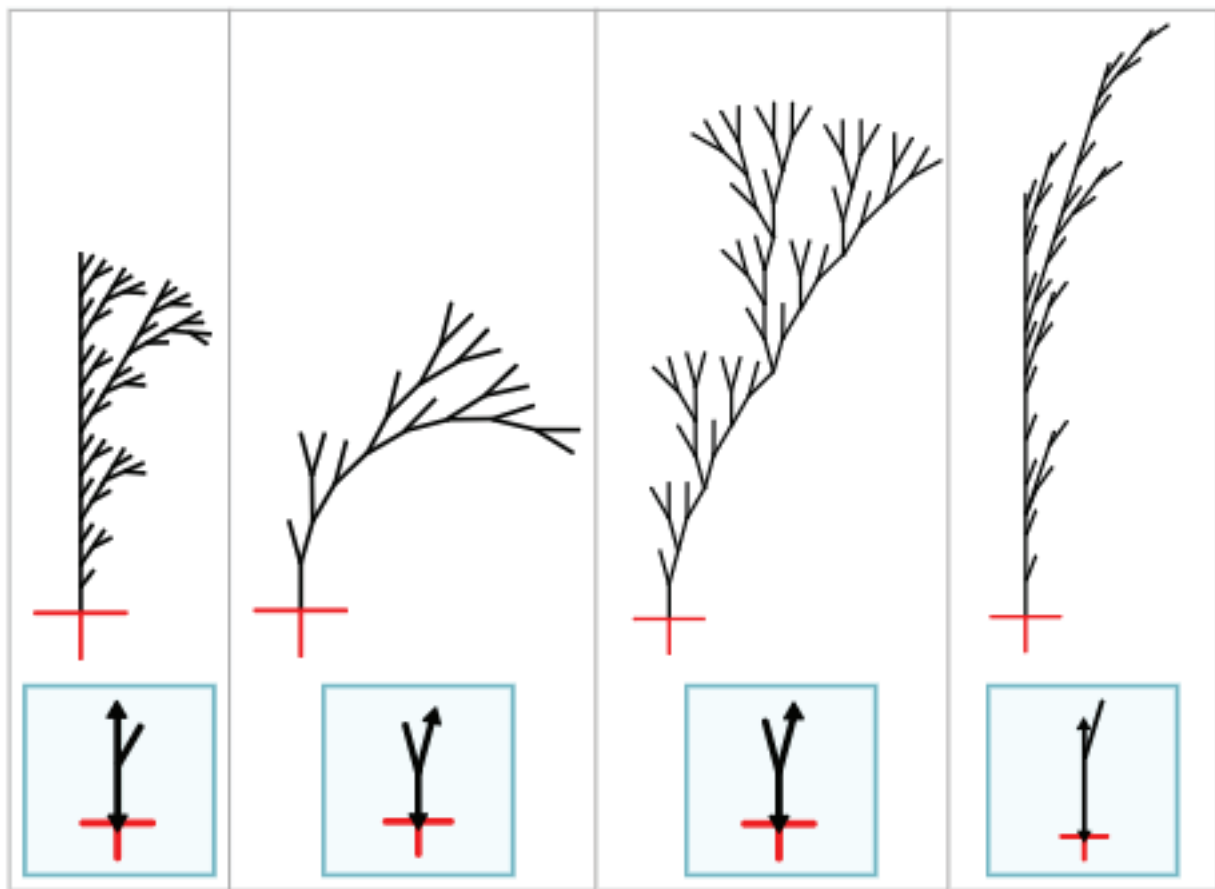
Answer - continued

Rule number 3:

At each iteration, the rest of the tree is added at the centre branch and the right hand branch can be longer than the centre branch. This only matches result number 4.

Rule number 4:

At each iteration, the rest of the tree is added to the right sub-branch and it is not pointing straight up but leaning to the right. The result must be result number 2.



It's Computational Thinking

Computational Thinking Skills: Decomposition, Pattern Recognition, Modelling and Evaluation, Algorithms, Evaluation

Concepts: Data Interpretation, Specification, Algorithms, Digital Systems

Lindenmayer Systems also known as L-Systems, invented by Aristid Lindenmayer, are a powerful method of describing the behaviour of plant cells and modelling the growth processes of plant development. L-systems can also be used to generate self-similar fractals.

What we have shown is only a very simplistic example of an L-System. This system is context-free, deterministic (not stochastic), there is only one rule and only one symbol.



Letter Code

In a prefix cipher, letters are transformed into a code made up of one or more digits. No code starts with the digits of another code. For instance, if the letter A is transformed into 12, the letter B can be transformed into 2 (because 12 does not start with 2).

In this case, the letter C can be transformed into 11 (because both 12 and 2 do not start with 11), but not into 21 (because this code starts with '2' which is the code for B) or into 121 (because this code starts with '12' which is the code for A).

Question

Click on the empty spaces below to add parentheses to the code in such a way that the digits enclosed in parentheses could represent the letters of the word BEBRAS.

(1 2 1 1 2 2 3 3 3 2 1)

Answer

Starting from the beginning, if B is encoded with two characters (12), then letter E has code 1 which is not possible because the code for B starts with the code of E. It is impossible for B to have a code longer than 2 (121, 1211, 12112, etc.) because this letter will repeat, but there are no repetitions of these sequences. **Thus the code for B is 1.**

The letter E is followed by a B, thus it can only be encoded as 2, 21 or 211223332. It cannot be 2 because then the word would start with BEBB. It cannot be 211223332 because then the word would be BEB. Thus the code for E is 21. At this point we know that 1_21_1 is the encoding for BEB and we need to put spaces in 2233321.

Consider now the letter S at the end of the word: its encoding cannot be 1 or 21 because these codes are already in use for B and E. So, the possible codes for S are 321, 3321, 33321, 233321 and 2233321. S cannot be encoded as 2233321 because then the word would be BEBS. It cannot be 233321 because there would be only one digit in order to encode the letters R and A. It cannot be 33321 because the equal digits in the sequence 22 would encode the different letters R and A. If S would be encoded as 3321, the letters RA would be encoded as 223. However, the code for R can't be just 2 and the code for A can't be just 3 because other codes start with that digit. Thus the code for S has to be 321, and 2233 encodes RA. As before, the code for R must be 22 and the code for A must be 33.

It's Computational Thinking

Computational Thinking Skills: Pattern Recognition, Algorithms, Evaluation

Concepts: Data Representation, Data Interpretation, Specification, Algorithms, Implementation, Digital Systems

The prefix cipher described in the task is an example of prefix code, in which objects are transformed into codes having the property that none of them can start with another code. This property of prefix codes is useful in information decoding, because there is no need to have separators between individual codes.

One common task using prefix codes is to find a prefix code for a given text, which makes encoded messages as short as possible. This is a task related to archiving data, i.e. decreasing its size while keeping all its information. A method to determine an optimal prefix code is provided by the Huffman coding. It is widely used, and is even used in popular media formats like JPEG and MP3.



Rabbit Paddock

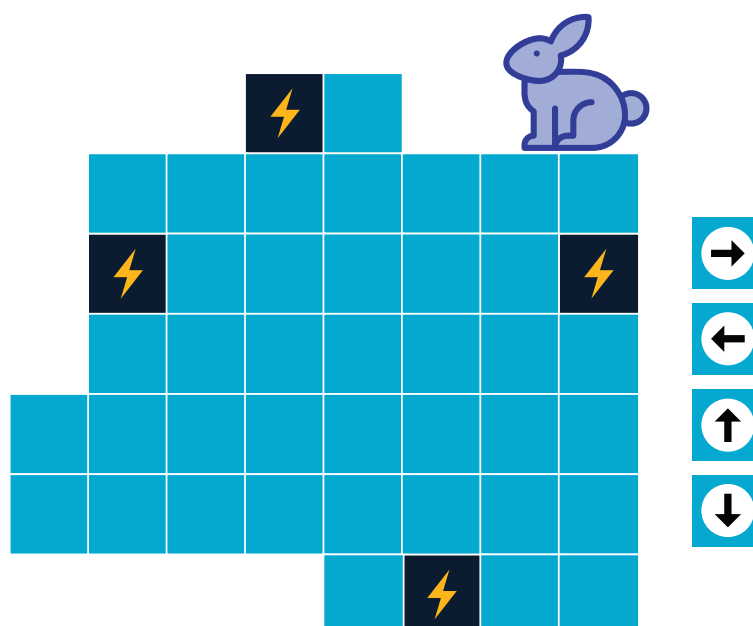
Jirka is playing with four robots on the grid shown that consists of regular tiles (blue) and charging station tiles (black). The robots move according to a program built using these commands:

- **STEP** – move forward to the next tile in front of you
- **LEFT** – turn 90 degrees anticlockwise, don't move forward
- **RIGHT** – turn 90 degrees clockwise, don't move forward

Question

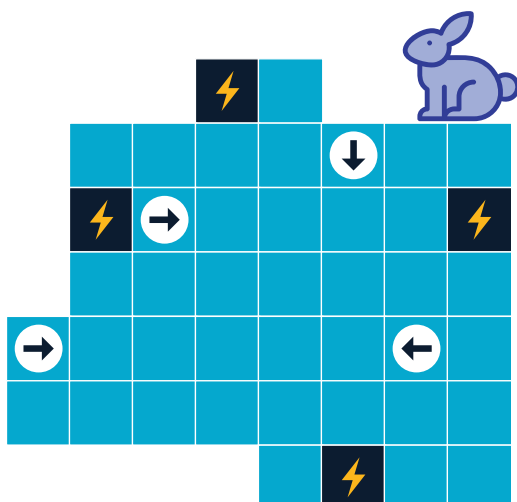
Jirka wrote a program **STEP LEFT STEP STEP** and downloaded it to all four robots.

Place four robots on the blue tiles and set their directions so that after executing the program, each of them ends up at a different charging station without crashing into each other.



Answer

The correct answer, graphically showing the correct positions and orientations, is:

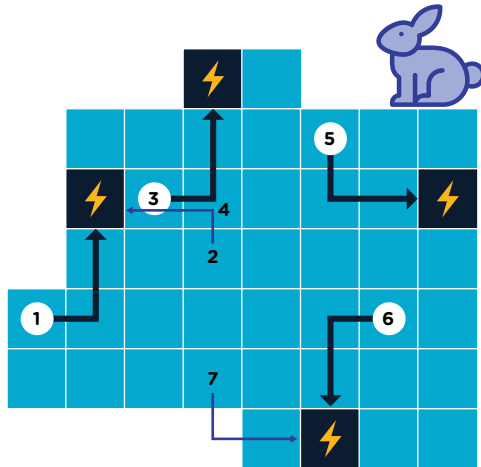




Rabbit Paddock - continued

Answer - continued

The correct solution is as follows.



Note, during the execution of the program, each robot travels along a path that is the shape of a rotated reflected character L, starting from its shorter “leg”.

We will use the table below with numbered cells in our explanation. The charging stations will be lettered A-D from left to right.

The charging station B is reachable only from the cell 3 because starting from cell 5 a robot crashes into a wall.

Cell 6 could be a starting point for charging stations C or D. However, while charging station C is reachable from cells 5 and 6, charging station D is reachable only from cell 6 (because starting from cell 7, the robot crashes into a wall). So cell 6 must be used as the starting point for charging station D. This means that cell 5 must be the starting point for charging station C.

There are two cells where a robot could start to reach charging station A: 1 and 2. If a robot starts from cell 2 and another robot starts from cell 3 at the same time, they crash each other in cell 4 after executing of first step of the program. So charging station A is reachable only from cell 1.

It's Computational Thinking

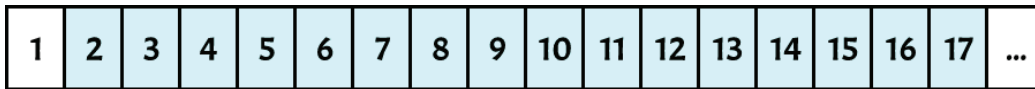
Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Digital Systems, Interactions

When creating a program, programmers must think not only about the intended function of the program, but must also consider situations that will cause the program to a crash. Such situations can be caused by a program running out of memory during a calculation, or fighting for a shared resource with another program. So, they must set initial conditions to make sure that program will be protected from crashes, or can avoid crashes. This may involve simulating the program, such as NASA simulating all instructions before sending them to the Mars Rover robot vehicles that have been on Mars since 2004.

Programming computers to do their work in any order, including in parallel, is called concurrent programming. This can be more difficult to manage when several different actions may be performed at the same time by different programs. In this task we looked at a simplified case where each robot executed the exact same program.

Reversibility



A robot can move objects that lie on a long surface, as in the figure above. The surface is divided into infinite numbered slots, and the number of a slot identifies the object's position. The robot is programmed by specifying some rules, and it moves the objects accordingly. After applying a rule, the robot cannot remember the position the object was in before it was moved.

An object is placed in one of the numbered slots. The robot can receive a MOVE or BACK command.

When the robot receives the command "MOVE rule 1", it executes rule 1.

When the robot receives the command "BACK rule 1", it reverses the effect of rule 1.

The BACK command can only be immediately applied after a MOVE command, which means the most recent MOVE.

A rule is considered reversible when the robot can execute MOVE and then BACK, of the same rule, without any confusion about what to do. Not all rules can be reversed.

Example

Rule A: Move the object to the slot on the right.

When the robot receives the command "MOVE rule A", it will move the object to the right. When the robot receives the command "BACK rule A", it will move the object to the left. So Rule A can be reversed.

Rule B: Move the object to slot 1.

When the robot receives "MOVE rule B", it will move the object to slot 1. When the robot receives "BACK rule B", it cannot determine where to place the object other than slot 1. So rule B cannot be reversed.

Consider the following rules:

1. If the object is in a position greater than 8, then move it to the slot on the right. If not, leave it where it is.
2. If the object is in a position greater than 8, then move it to the slot on the left. If not, leave it where it is.
3. If the object is in an even position, then move it 2 slots to the right. If not, move it to the slot on the left.
4. If the object is in an even position, then move it 2 slots to the right. If not, move it 2 slots to the left.

Question

Select all of the above rules that can be reversed.

Rule 1

Rule 2

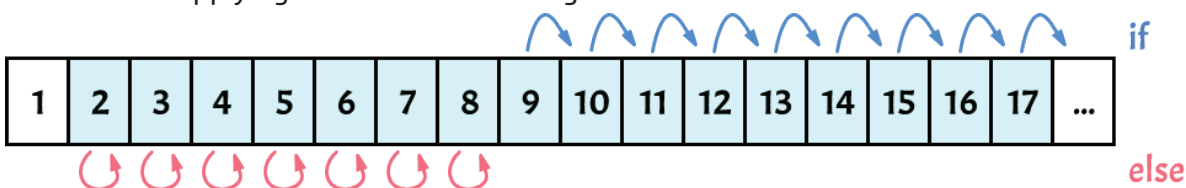
Rule 3

Rule 4

Answer

The correct answers are rules 1 and 4.

Let us number the slots from left to right starting from 1, and consider first rule 1 and rule 4, one at a time. The effect of applying rule 1 is the following:



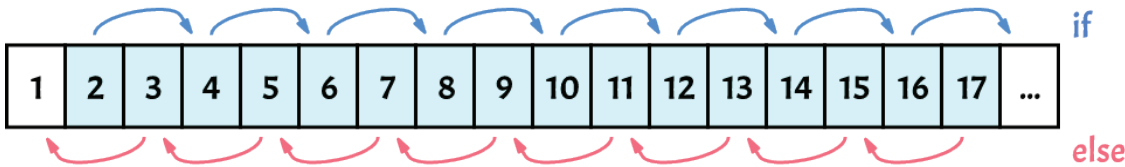
If the object was in a position greater than 8, then it will end up in a position that is greater than 9 (blue arrows); if the object was in a position lower than 9 (i.e., up to 8 included), then it will not be moved and hence it will still be on a position lower than 9 (red arrows). (Notice that the object cannot end up in position 9.) So, it is easy to distinguish the two cases and to move back the object accordingly, that is, rule 1 can be reversed by this rule:

Continued on next page

Reversibility - continued

If the object is on a position greater than 9, then move it to the slot at the left, otherwise leave it where it is.

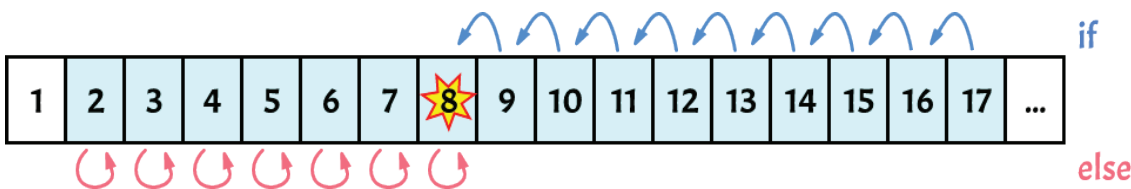
The effect of applying rule 4 is the following:



If the object was in an even position, it will end up in another even position (blue arrows), whereas if the object was in an odd position, it will end up in another odd position (red arrows). Hence it is always possible to distinguish the two cases and to move back the object accordingly, that is, rule 4 can be reversed as follows:

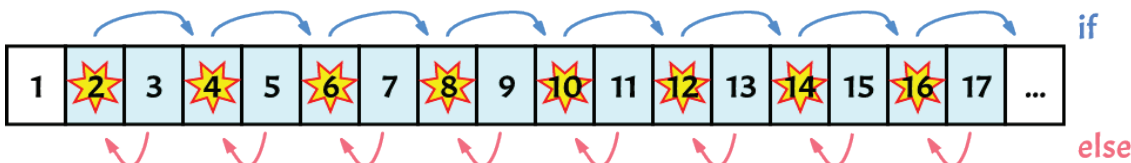
If the object is in an even slot, then move it 2 slots to the left, otherwise move it 2 slots to the right.

Let us now explain why the other rules cannot be reversed. Rule 2 has the following effects: if the object was in position 8, then it will not be moved hence it will stay in position 8 (red arrow); if the object was in position 9, then it will end up in position 8 (blue arrow).



So, after the application of this rule, if the object is in position 8, one cannot establish whether it has been moved or it was already there, that is, the effect of rule 2 cannot be reversed.

The effect of applying rule 3 is the following: if the object was in an even position, it will end up in another even position (blue arrows); if it was in an odd position, it will end up in an even position (red arrows).



So, after the application of this rule, one cannot distinguish between the two cases and cannot recover the original position. That is, the effect of rule 3 cannot be reversed.

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Implementation, Digital Systems

The rules presented in the task have the form if ... then... else. In Computer Science this is called a conditional statement; the term is due to the fact that this kind of statement specifies to do something only when some condition (also called test) holds true. The conditional statement is one of the basic features of programming languages, that are used to program the behaviour of automatic devices, like the robot in this task, or a computer. If a program contains a conditional statement, its execution may follow two different flows; indeed, according to the initial conditions, or inputs, the condition may be true or false and hence different commands are executed.

A conditional statement is reversible if one can undo its effect, i.e. one can restore the initial situation starting from the outcome. In general, the effect of a conditional statement is not reversible, because the outcome may not be enough to restore the initial situation. In this task, in order to decide which conditional statements are reversible, one needs to consider all possible final outputs and verify if some "overlaps" may occur, as for rules 2 and 3. Reasoning about the effects of programming statements and their properties, like reversibility, is an important ability for programmers.



Musical Instrument

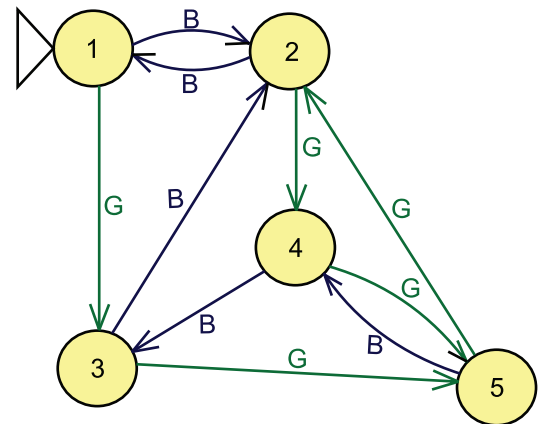
Mr. Beavo has created a special musical instrument. There are only three keys for producing sound. They are the red key (R), the blue key (B), and the green key (G). It can produce 5 different notes: 1, 2, 3, 4, and 5.

To start playing the instrument, R should be pressed firstly, and it will sound Note 1 as the initial note.

After that, the instrument will sound a note based on the previous note and the key being pressed. The “note change diagram” shows it.

At any time, when R is pressed, it will sound note 1. A song is a sequence of notes. A sequence is well-formed if it ends with two times Note 1.

For instance, the sequence “R-G-B-B-R” will make the instrument produce the notes 1-3-2-1-1, but the sequence “R-G-B-R-B” will make the instrument produce the notes 1-3-2-1-2.



Question

Which from the following sequences is well-formed?

R-B-B-G-B-R

R-G-G-G-B-R

R-B-G-B-G-R

R-G-G-B-G-R

Answer

The correct answer is B.

Basically, the answer can be found by simulating the instrument behaviour. You can follow the arrows on the diagram. But you can analyse it first: in each sequence we pressed the button R (red) two times, at the beginning and at the end. There is no arrow on the diagram that points to 1 and is green (i.e. G, when the green button is pressed). So you can exclude C and D, because those sequences don't end by two times Note 1.

The first sequence (answer A) will play Note 2 last, before pressing the last R. So it is not a well-formed sequence either. The last remaining possibility is answer B, which we can verify will play these notes: 1-3-5-2-1-1.

It's Computational Thinking

Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Interpretation, Specification, Algorithms

This problem's computation model is the finite-state machine. Many problems, some simple, some very complicated, can be solved in Computer Science with the help of the finite-state machines (also called finite automata). For instance, they are often used by programming-language compilers — programs that read and verify the validity of computer code.

A good example of a finite-state machine in the physical world is a coffee-to-go machine. Before it makes coffee, it needs to reach the state where we have paid enough and inserted enough coins. So, the insertion of a coin may have a different effect on the machine: either count up what has been paid and wait, or indicate that enough has been paid and make coffee. Same action, different effect — depending on what has been done before, just like the musical instrument described in this task.



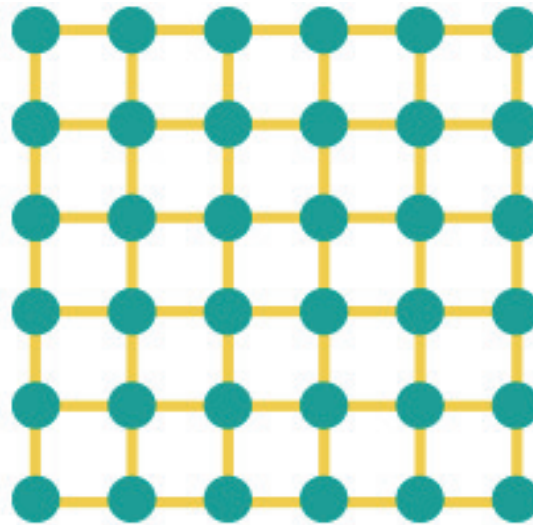
Royal Fountains

King Beaver is visiting the royal fountains, shown by green circles below. He has a request:

"I wish to go on a walk from one fountain to another fountain. I will tell you which two when I arrive. I don't care how long the walk is, but do I expect to be able to walk on red carpets. I do not care if the first path from my start fountain or the last path to my final fountain has a red carpet or not."

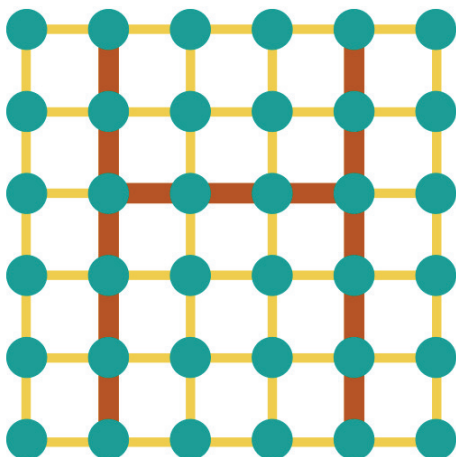
Question

Click on the paths between the fountains to lay down the fewest number of red carpets so that the king's request will be honoured.



Answer

The minimum number of carpets is 13.



The image shows a possible solution. In this case, the strategy was to start by drawing vertical lines. Since we have six columns and we may leave one path without red carpet, two vertical lines are enough to enable, at least, one connection between all the fountains.

However, this is not enough. If we want to cross from one side of the garden to the other, we need to connect the vertical lines. Otherwise, it is not possible to do movements such as going from the fountain in the top left corner to the fountain on the bottom right corner. Thus, we need to add an horizontal line that connects both vertical lines.

In the end, we have five paths in the first vertical line, another five paths in the second vertical line, and three paths in the horizontal line. Therefore, we need a minimum of 13 paths. There are in fact 12 possible ways of arranging the carpets though.

It's Computational Thinking

Computational Thinking Skills: Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Digital Systems

To find a minimal number of essential connections in a structure like a street map is a problem one often faces, for instance in planning cables for electricity or optical fibre. Graph theory provides a mathematical background for this kind of problem.



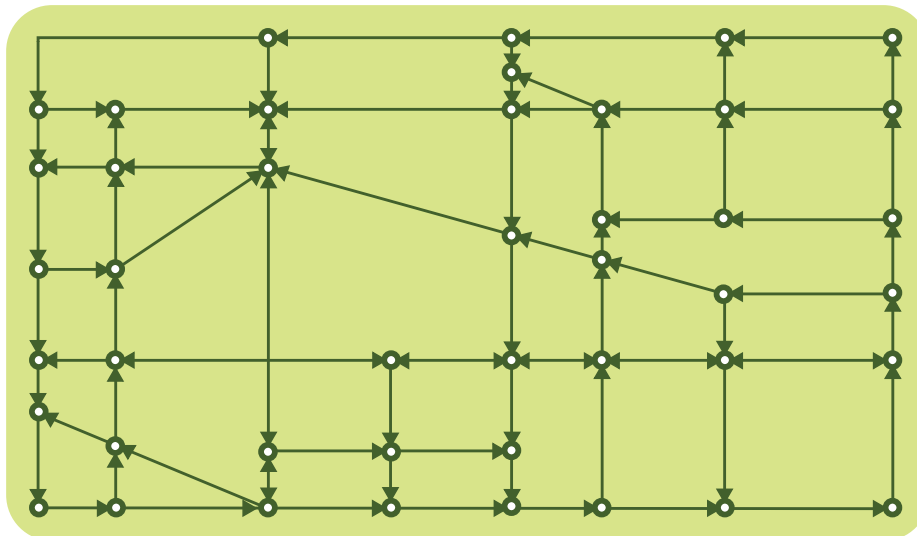
Vulnerable

In this city map you can see that many streets have one-way traffic, indicated by a single arrow-head.

Some of the streets have two-way traffic, indicated by two arrow-heads. An intersection is called vulnerable if it can only be reached from one other intersection.

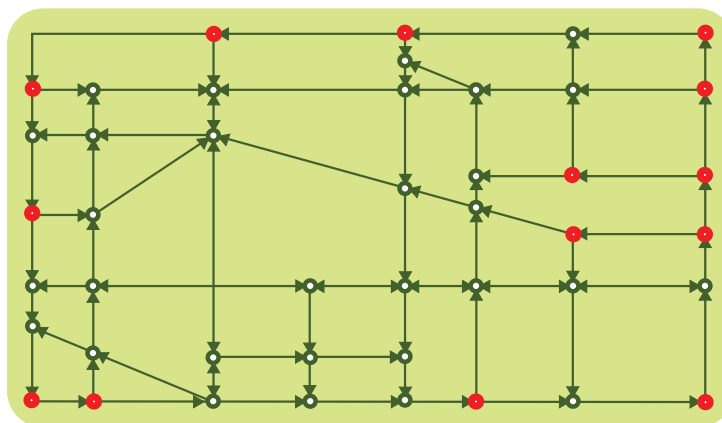
Question

Select the vulnerable intersections on the diagram below.



Answer

There are 14 vulnerable intersections in the map as marked in the picture.



It's Computational Thinking

Computational Thinking Skills: Pattern Recognition, Modelling and Simulation, Evaluation

Concepts: Data Collection, Data Interpretation, Specification, Algorithms, Digital Systems, Interactions, Impacts

A structure like this city map is called a network. An attack on a vulnerable spot in a network can create inaccessibility for certain spots. Analysing a network structure, looking for these dangerous places, is important in network design to reduce the loss of information that will be transmitted. A network is a directed graph and the number of arrows pointing towards a specific point is called the indegree of that point. A point is vulnerable if the indegree is 1 and inaccessible if the indegree is 0, though this is never the case in this network.

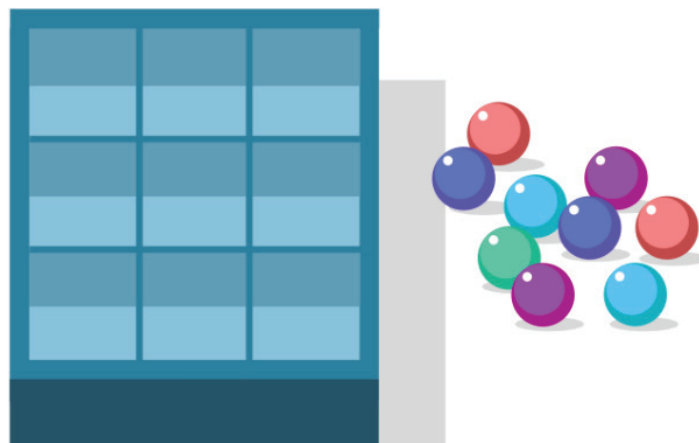


Marbles and Boxes

Hira has a box with 9 trays, as well as 9 marbles.

Hira chooses between 0 and 9 marbles and places them in the box according to the following rules:

- Each marble is in a different tray.
- The total number of marbles in each row is even.
- The total number of marbles in each column is even.



Question

In how many different ways can Hira place the marbles in the box?

12

16

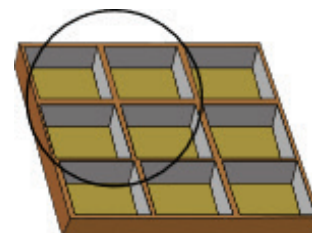
64

512

Answer

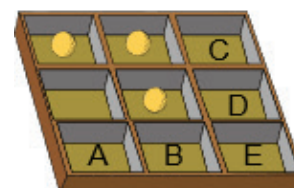
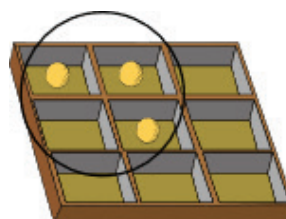
The correct answer is B) 16.

First consider just a 2x2 section of the box. There are 4 trays in this section and each tray will either have a marble in it or not. Therefore, there are $2^4 = 16$ different ways Hira can place marbles in this section.



An important observation to make is that after Hira decides how to fill this section, Hira no longer has any choice regarding how to fill the remaining row and column. For each remaining tray, the requirement that the row totals and column totals must be even will force Hira to either include a marble or not.

For example, suppose Hira fills the section as shown: Since the first column only has 1 marble, Hira must place a marble in tray A to make the column total even. Column 2 already has an even number of marbles so Hira must leave tray B empty. Using similar reasoning, Hira must leave C empty and place marbles in D and E. Therefore, Hira can place the marbles in the box in 16 different ways.



It's Computational Thinking

Computational Thinking Skills: Decomposition, Modelling and Simulation, Algorithms, Evaluation
Concepts: Data Representation, Data Interpretation, Specification, Algorithms

Transmitting data both safely and accurately is an important Computer Science task. One way to make sure that data has not been lost or changed during transmission is to perform a parity check. A parity bit is added to the end of the original message based on the contents of the original message. When the message is received at its destination, the parity bit is recalculated and if it doesn't match the parity bit received, it serves as evidence that the message has been corrupted during transmission.

In this task, the final row and final column serve as parity bits. If the box of marbles was sent as a message, the receiver could check the totals and if they are not even, report back to Hira that something has gone wrong.



Blinking LEDs

You received a programmable electronic board and started to play with it. On this board, there are three LEDs (one red, one green and one blue) — a particular kind of light device — which you can control with a program by turning them on or off (they are all off before the program starts). Here is an example of such a program:

REPEAT:

```
| turn_on (RED_LED);  
| wait (1s);  
| turn_off (RED_LED);  
| wait (2s);
```

The actions performed by this program are as follows:

1. turn on the red LED,
2. wait and do nothing for 1 second,
3. turn off the red LED,
4. wait and do nothing for 2 seconds,
5. and start again with step 1.

The red LED will blink forever, alternating between being on for 1 second and being off for 2 seconds.

You found the following program on internet and want to try it on your board:

REPEAT:

```
| turn_on (BLUE_LED);  
| wait (2s);  
| turn_on (RED_LED);  
| turn_on (GREEN_LED);  
| wait (2s);  
| turn_off (GREEN_LED);  
| turn_off (BLUE_LED);  
| wait (2s);  
| turn_on (GREEN_LED);  
| wait (2s);  
| turn_off (RED_LED);  
| turn_off (GREEN_LED);
```

Question

How many LEDs are on 13 seconds after the above program has been started?

0

1

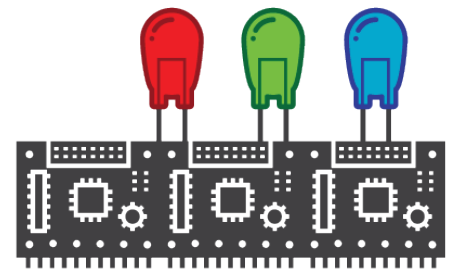
2

3

Answer

The correct answer is 1.

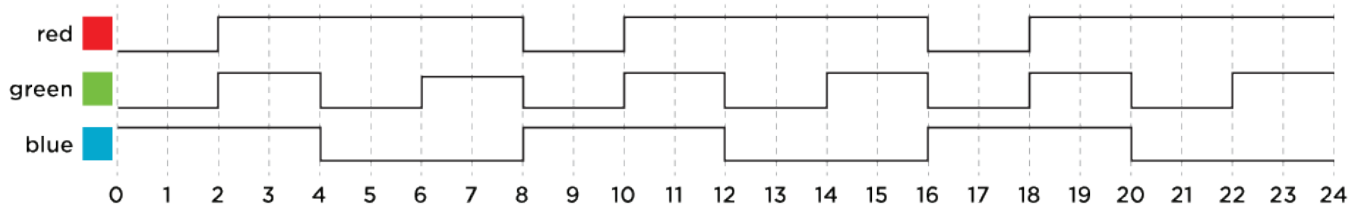
The diagram on the following picture shows the status of the three LEDs over time. On this diagram, horizontal lines can either be drawn high (1) or low (0), with these two states respectively corresponding to the LED being on or off. After 13 seconds, the red LED is on (since the appropriate line is in the 1 position) and the green and blue ones are off (since their lines are in the 0 position).





Blinking LEDs - continued

Answer - continued



It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Data Interpretation, Specification, Algorithms, Implementation, Digital Systems

Being able to understand a program is an important part of informatics. In this task, a particular style of programming known as procedural programming is used. It's the kind of programming used with programmable electronic boards (such as an Arduino electronic board), which is a very popular way of learning programming.

In this particular task, the learner has to read and understand a simple program and report the status of three LEDs over time, that is, simulate and trace the execution of the program. This is an important activity that programmers also have to do when something does not work as the programmer thought it would, and is a part of what we call debugging. Debugging requires a programmer to understand what a program is doing, in order to discover the cause of something that is going wrong with the program.

Game of Whispers

Five friends sit in a sequence: Anjali, Bernard, Chandra, Damini, and then Eshwar. Anjali whispers to Bernard, spelling out a ten letter word (e.g. V-I-S-I-B-I-L-I-T-Y).

Bernard whispers to Chandra, spelling out the same word, but with one error.

The error could be a replacement of a letter with a new letter (e.g. V-I-Q-I-B-I-L-I-T-Y) or the deletion of a letter (e.g. V-I-S-I-B-I-L-I-Y).

Chandra, in turn, whispers the spelling to Damini with one error and so on.



In each whisper, there is exactly one error — no more, no less — but the same letter or position could be involved in more than one error in the chain.

Question

If the spelling whispered by Anjali to Bernard is A-D-V-E-N-T-U-R-E-S, which of these spellings could be whispered to Eshwar? [Select all that apply]

A-D-E-N-U-R

A-D-D-E-N-T-U-R-E-S

A-D-V-E-N-T-U-R-E

A-V-E-N-G-E-R-S

D-E-N-T-U-R-E

Answer

Correct answers: options B, C and E.

A= A-D-E-N-U-R

B= A-D-D-E-N-T-U-R-E-S

C= A-D-V-E-N-T-U-R-E

D= A-V-E-N-G-E-R-S

E= D-E-N-T-U-R-E

Anjali whispers the correct spelling to Bernard. Bernard whispers the spelling with one error to Chandra. Chandra whispers to Damini with one more error. And finally, Damini whispers to Eshwar with one more error. So there are exactly three errors that take place by the time the spelling is whispered to Eshwar.



Game of Whispers - continued

Answer - continued

Option A) Incorrect. There are only six letters in this option, indicating that four letters have been deleted. But there are a maximum of three total errors.

Option B) Correct. V is replaced by D in one error. Then, any letter (say T) is replaced in the second error. That letter is replaced again in the third error by the original letter (T). i.e. A-D-V-E-N-T-U-R-E-S \rightarrow A-D-D-E-N-T-U-R-E-S \rightarrow A-D-D-E-N-C-U-R-E-S \rightarrow A-D-D-E-N-T-U-R-E-S.

Option C) Correct. S is deleted. As with Option B, any other letter can then be replaced twice, coming back to the original letter. i.e. A-D-V-E-N-T-U-R-E-S \rightarrow A-D-V-E-N-T-U-R-E \rightarrow A-D-V-E-N-C-U-R-E \rightarrow A-D-V-E-N-T-U-R-E.

Option D) Incorrect. The option has only 8 letters. That means two deletions have taken place. Further, this option has two new letters (G and E) indicating that two replacements have taken place as well. This means that at least 4 errors have taken place.

Option E) Correct. This option needs exactly three deletions (A, V, and S).

It's Computational Thinking

Computational Thinking Skills: Decomposition, Abstraction, Modelling and Simulation, Algorithms, Evaluation

Concepts: Abstraction, Specification, Algorithms, Digital Systems

This task is an example of the concept of noise in information transfer. The noise in this question is deletion and replacement of letters. We know that the noise here is restricted to three letters. So we can be sure that the final word reaching Eshwar should have a minimum of seven letters which match (in sequence) with the original word.

A more real world application for such noise in information transfer comes in transmitting digital signals composed of bits i.e. signals consisting of only 0's and 1's. Such signals could have the same errors of replacement and deletion. However, in these cases, replacement simply means switching the bit i.e. 0 becomes 1 and vice versa.

Claude Shannon, who is famously known as the 'Father of Information Theory', did some groundbreaking work on how to send a digital signal with almost zero error where he discusses the impact of the channel through which the signal is being sent.

Curriculum and Computational Thinking skills alignment by Allira Crowe and Graeme Buckie.

This Solutions Guide was created by Hannah Piper and edited by Ruwan Devasurendra.

We would like to thank the International Bebras Committee and community for their ongoing assistance, resources and collaborative efforts.

Special thanks to Eljakim Schrijvers, Alieke Stijf and Dave Oostendorp for their support and technical expertise.

If you would like to contribute a question to the International Bebras community, please contact us via the details below.

Contact us

CSIRO Digital Careers

digitalcareers@csiro.au

digitalcareers.csiro.au